Санкт-Петербургский государственный университет

Прикладная математика и информатика

Статистическое моделирование

Ершов Василий Алексеевич

Исправление ошибок в чтениях, полученных с помощью технологии IonTorrent

Выпускная квалификационная работа

Научный руководитель:

к.ф.м., доцент А. И. Коробейников

Рецензент:

Разработчик ПО А. Л. Тарасов

Санкт-Петербург

2017

Saint Petersburg State University
Applied Mathematics and Computer Science
Statistical Modelling

Ershov Vasilii Alekseevich

Read error correction for IonTorrent data

Graduation Project

Scientific Supervisor:
Associate Professor A. I. Korobeynikov

Reviewer:
Software Developer A. L. Tarasov

Saint Petersburg
2017

Оглавление

Введение . . . 5

Глава 1. Постановка задачи и профиль ошибок технологии секвенирования IonTorrent . . . 7

1.1. Основные обозначения . . . 7

1.2. Задача исправления ошибок . . . 8

1.3. Профиль ошибок . . . 8

1.4. Quality-значения технологии IonTorrent . . . 9

Глава 2. Метод исправления ошибок . . . 13

2.1. Основная идея алгоритма BayesHammer . . . 13

2.2. IonHammer . . . 15

2.3. Алгоритм оценки множества геномных hk-меров . . . 16

2.3.1. Статистики по встретившимся hk-мерам . . . 16

2.3.2. Кластеризация ED_l-графа . . . 18

2.3.3. Субкластеризация . . . 21

2.3.4. Отличия от предыдущей версии IonHammer . . . 23

2.3.5. Качество алгоритма кластеризации . . . 24

2.4. Алгоритм коррекции ошибок . . . 25

2.4.1. Алгоритм коррекции . . . 25

2.4.2. Функция штрафа . . . 29

Глава 3. Оценка качества алгоритма . . . 32

3.1. Сравнение новой версии IonHammer со старой . . . 32

3.2. Сравнение IonHammer с другими алгоритмами коррекции . . . 34

Заключение . . . 38

Список литературы . . . 39

Приложение А. Псевдокод алгоритмов . . . 41

Приложение Б. Наборы данных . . . 44

Приложение В. Сравнение времени работы алгоритмов 45

Приложение Г. Таблицы с качеством коррекции чтений 47

Приложение Д. Результаты сборки генома 53

Введение

Совокупность наследственного материала, заключенного в клетке организма, называется геномом. В геноме содержится биологическая информация, определяющая развитие организма. Обычно геномы живых организмов построены на основе макромолекул дезоксирибонуклеиновой кислоты (ДНК). Изучение ДНК позволяет решать широкий спектр задач, среди которых:

- История развития жизни на планете.
- Выявление причин и лечение передающихся по наследству заболеваний.
- Поиск новых антибиотиков и других лекарств.
- Различные задачи генной инженерии (например, разработка генно-модифицированных продуктов).

Для изучения ДНК ее требуется преобразовать из макромолекулы в удобный для анализа формат — строчку над алфавитом $\{A,\ C,\ G,\ T\}$. Процесс преобразования макромолекулы в строчку называется секвенированием ДНК. Это сложная задача и на сегодняшний день не существует метода, позволяющего получить полную цепочку ДНК. Вместо этого, существующие технологии читают много небольших участков ДНК, на основе которых с помощью специальных алгоритмов затем восстанавливается полная цепочка. К сожалению, секвенированные участки содержат ошибки, что усложняет, а иногда и делает невозможным восстановление полной цепочки и возникает необходимость эти ошибки исправлять. Как и в большинстве задач, универсального «инструмента», позволяющего исправлять ошибки для разных технологий секвенирования не существует — в разных технологиях совершаются разные ошибки и требуются различные подходы к их исправлению.

Одной из самых распространенных технологий секвенирования является технология Illumina. Для данной технологии основой тип ошибок — замена одного нуклеотида на другой. Для коррекции ошибок такого рода существует алгоритм алгоритм BayesHammer [1], являющийся частью геномного ассемблера SPAdes [2]. Кроме того, в SPAdes реализован алгоритм коррекции ошибок вида «вставка» или «удаление» IonHammer, предназначенный для исправления ошибок, возникающий при секвениро-

вании с помощью технологии IonTorrent. Данная версия алгоритма коррекции обладает несколькими недостатками, из-за которых алгоритм может работать достаточно долго, а также исправляет небольшое число ошибок.

В данной работе предложена и реализована модификация алгоритма IonHammer, а также исследовано качество и время работы новой версии алгоритма.

Глава 1

Постановка задачи и профиль ошибок технологии секвенирования IonTorrent

1.1. Основные обозначения

Пусть задан алфавит $\mathbb{A} = \{A,\ C,\ G,\ T\}$ из нуклеотидов, содержащихся в ДНК организма. Введем несколько определений, которые понадобятся в дальнейшем.

Определение 1. k-мером будем называть элемент $\mathbb{A}^k$ — строчку из k символов над алфавитом $\mathbb{A}$. Множество всех возможных строчек из алфавита $\mathbb{A}$ будем обозначать $\mathbb{A}^* = \bigcup\limits_{i=1}^{\infty} A^i$.

Определение 2. Будем называть гомополимером пару (n, l), где $n \in \mathbb{A}$ — нуклеотид, а $l \in \mathbb{N}_{\geq 0}$ — длина гомополимера. Множество таких пар будем обозначать за $\mathbb{H}$. Если явно не указано иного, то мы считаем, что длина гомополимера больше 0 (в некоторых ситуациях нам будет удобно предполагать, что длина гомополимера бывает равна 0).

Определение 3. hk-мером будем называть элемент $\mathbb{H}^k$. Множество всех возможных строчек из гомополимеров будем обозначать за $\mathbb{H}^*$.

Определение 4. Введем обозначения для операций над строками:

1. $x[k : m)$ — подстрока x с k-ого до $m - 1$ символа включительно.
2. $x[k : m]$ — подстрока x с k-ого до m символа включительно.
3. $x \,|\, y$ — конкатенация строки x и строки y.

Обозначения в зависимости от контекста используются для строк из нуклеотидов или для строк из гомополимеров.

Замечание. Соответствие между hk-мерами и элементами $\mathbb{A}^*$ не является взаимно-однозначным, поэтому в дальнейшем мы будем считать, что у любых двух соседних гомополимеров будут различные нуклеотиды, т.е. для любого $s \in \mathbb{H}^k$ и для любого $i \in 1 \ldots k - 1$ основание $s[i]$ не равно основанию $s[i + 1]$. Также под hk-мером мы будем понимать как последовательность элементов $\mathbb{H}$, так и строчку, порождаемую данным hk-мером.

Определение 5. Под расстоянием Левенштейна двух строк x и y будем понимать минимальное количество операций вставки одного символа, удаления одного символа и замены одного символа на другой, необходимых для превращения одной строки в другую [3]. Если не указано иного, то под расстоянием между строчками (в том числе и между hk-мерами) мы будем понимать расстояние Левенштейна.

Определение 6. Пусть s является оценкой подстроки g с позиции m до позиции k (т.е. мы допускаем, что в s могут содержаться ошибки). Тогда для любой позиции $i \in [m, k]$ мы будем говорить, что s покрывает позицию i.

1.2. Задача исправления ошибок

При секвенировании генома $g \in \mathbb{A}^*$ получается множество строк $\mathbb{S}$, элементами которого являются оценки подстрок генома g. Элементы множества $\mathbb{S}$ мы будем называть *чтениями*. Чтения обычно содержат ошибки. Задача исправления ошибок заключается в нахождении для каждого чтения $s \in \mathbb{S}$ подстроки генома g, в результате секвенирования которой была получена строка s. При этом сам геном g не известен.

В общем случае такую задачу решить нельзя — если каждый символ генома покрыт только одним чтением, то понять, есть ли ошибка и как ее исправлять не представляется возможным. Современные секвенсоры генерируют миллионы строчек, в результате чего практически для каждой позиции i из генома g существует несколько чтений $s \in \mathbb{S}$, покрывающих данный символ. Таким образом возникает возможность использовать статистические методы для исправления ошибок.

1.3. Профиль ошибок

Для различных технологий секвенирования известен профиль ошибок, который связан с физическим и/или химическим принципом работы прибора, производящего секвенирования. Например, для технологии секвенирования Illumina почти все ошибки — замены одного нуклеотида на другой. С таким профилем ошибок относительно просто работать и подавляющее большинство алгоритмов коррекции предназначено для исправления ошибок такого рода. Для технологии секвенирования IonTorrent ошибки устроены более сложным образом — участок генома читается не посимвольно, а целыми гомополимерами. В первом приближении, достаточном для разработки

алгоритмов коррекции, IonTorrent при чтении участка генома генерирует последовательность пар (n, l), где $n \in \mathbb{A}$, а $l \in \mathbb{R}_{\geq 0}$ — некоторая вещественная оценка длины гомополимера. Длина l может рассматриваться как случайная величина со средним, равным длине гомополимера в геноме и небольшой дисперсией. Затем для каждого элемента последовательности длина l округляется до ближайшего целого. Полученная последовательность пар $(n, [l])$ преобразуется в строку $s \in \mathbb{A}^*$. Из-за такого принципа работы большая часть ошибок в чтении s это вставки или удаления, возникшие в результате неправильного округления длины гомополимера. На рис. 1.2 изображены эмпирические частоты округленных длин l при условии настоящей длины гомополимера. Видно, что большая часть ошибок имеет размер один. Кроме того, вероятность ошибки увеличивается при увеличении длины гомополимера — такое поведение является известной особенностью технологии: IonTorrent не умеет измерять гомополимеры длины больше, чем $14 - 16$, а также для длин больше 7 допускает ошибки в подавляющем числе случаев.

На рис. 1.1 по оси y изображена доля от общего числа ошибок в зависимости от позиции символа в чтении. Из графика видно, что большая часть ошибок приходится на нуклеотиды в конце чтения. В данном примере при секвенировании использовался чип, позволяющий IonTorrent за раз считать строчки из примерно 400 нуклеотидов. В итоге видно, что после 400 прочитанных символов количество ошибок существенно возрастает и если примерно в этом месте выбрать границу, до которой брать нуклеотиды, то «качество» чтения можно улучшить, ценой потери некоторого количества информации.

1.4. Quality-значения технологии IonTorrent

Технология IonTorrent, как и многие другие секвенсоры, в результате секвенирования выдает не только последовательности нуклеотидов, но и дополнительную вспомогательную информацию о том, насколько «хорошо» прочитано чтение.

Каждой прочитанной строке $s \in \mathbb{S}$ сопоставлен вектор quality-значений (quality-статистик) $q_s \in \mathbb{R}^{|s|}$, где под $|s|$ мы понимаем длину чтения s. Элементы этого вектора характеризуют то, насколько секвенсор «уверен» в том, что i-ая позиция строки s прочитана без ошибок. Во многих технологиях секвенирования элементы вектора q являются

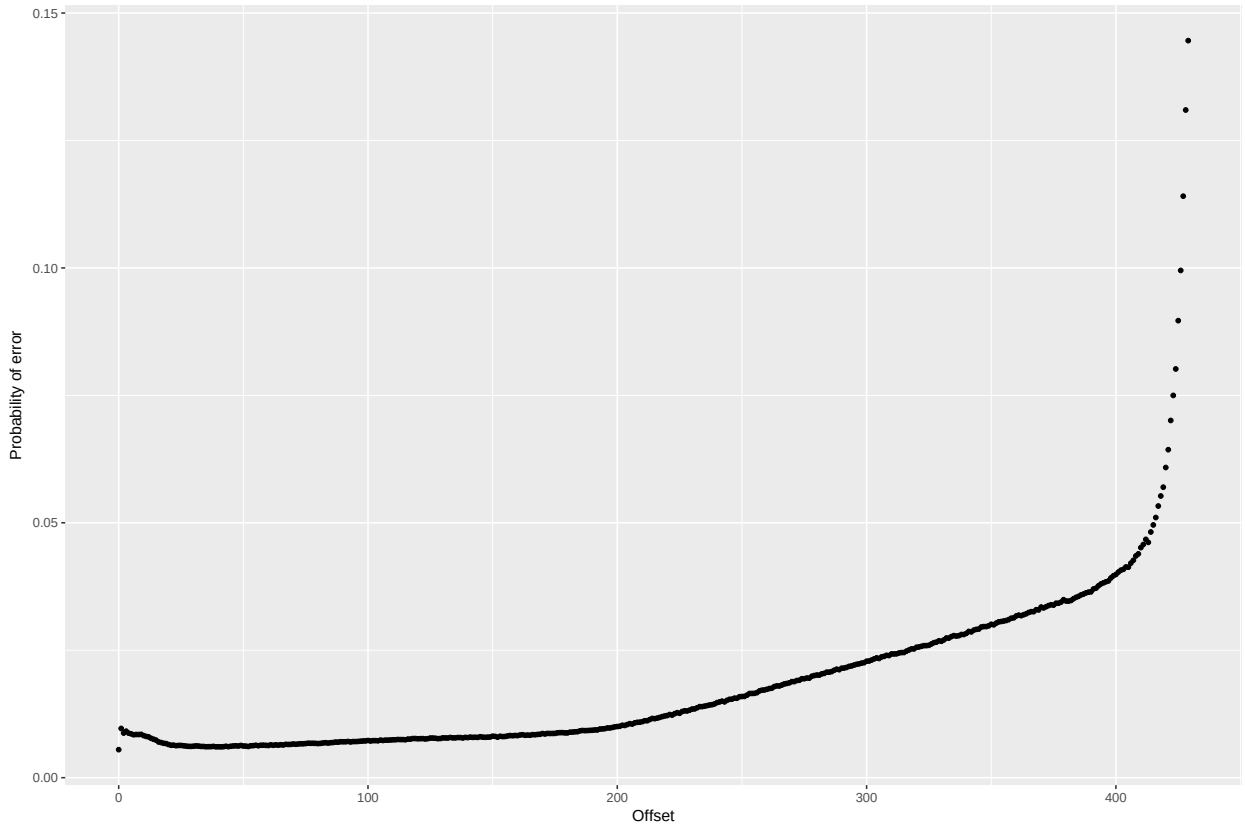

Рис. 1.1. Вероятности ошибок в зависимости от позиции. График построен на основе 7247730 чтений бактерии *E. coli str. DH10B*, секвенированной с помощью 520 чипа.

логарифмическим преобразованием оценки вероятности ошибки в данном символе:

$$q[i] = -10 \log_{10}(p[i]),$$

где $p[i]$ — оценка вероятности ошибки. Из-за особенности технологии IonTorrent не очень понятно, как понимать вероятность ошибки в данном символе, т.к. основная часть ошибок — вставки и удаления, а не замены. Тем не менее, на официальном сайте IonTorrent [4] утверждается, что данные числа все-таки являются оценкой вероятности ошибки в данном символе.

Замечание. В IonHammer мы для связи $q[i]$ и $p[i]$ используем натуральный логарифм:

$$q[i] = \log(p[i]),$$

где log – натуральный логарифм. В дальнейшем мы всегда будем предполагать такую связь, а не с основанием 10.

Мы проанализировали данные статистики на основе нескольких наборов данных и пришли к выводу, что трактовать $p[i]$ как вероятности ошибок нельзя. В тоже время,

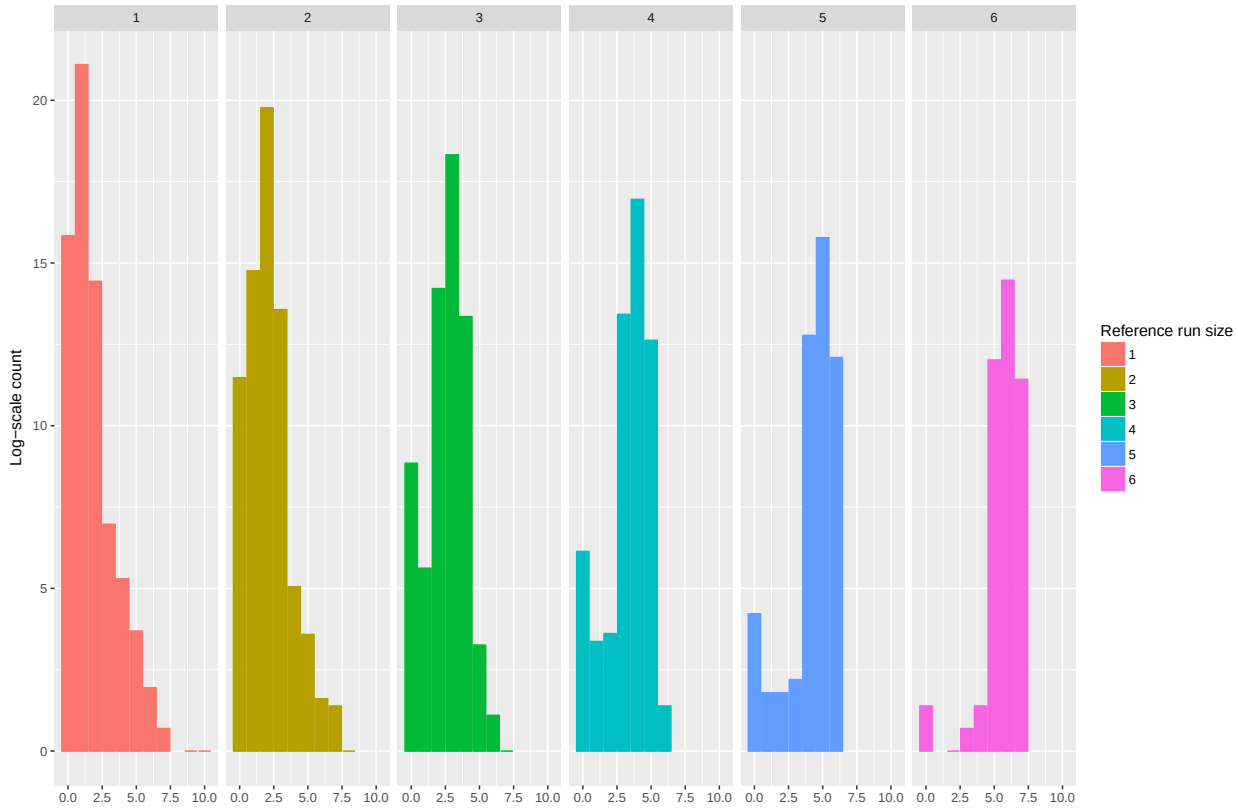

Рис. 1.2. Эмпирические частоты ошибок для различных длин гомополимеров. По оси y изображен логарифм частоты встречаемости оценки длины гомополимера.

данная величина сильно коррелирует с вероятностью ошибки в символе, как видно из графика на рис. 1.3 и, таким образом, может быть использована для построения статистик, характеризующих качество прочитанной подстроки.

Пусть дана строка s и соответствующие значения вектора q, а также соответствующий ему вектор оценок вероятностей ошибок p. Тогда мы можем определить качество прочтения строки s с помощью формулы 1.1. Отметим, что точно таким же образом можно определять качество прочтения любой подстроки s. Чем меньше $Q(s, q)$, тем лучше «качество прочтения» строки s. Если $p[j]$ являются хорошими оценками вероятностей ошибки, то $Q(s, q)$ можно трактовать, как оценку логарифма вероятности того, что s содержит ошибку.

$$Q(s, q) = \log\left(1 - \prod_{j=1}^{|s|}(1 - p[j])\right) \tag{1.1}$$

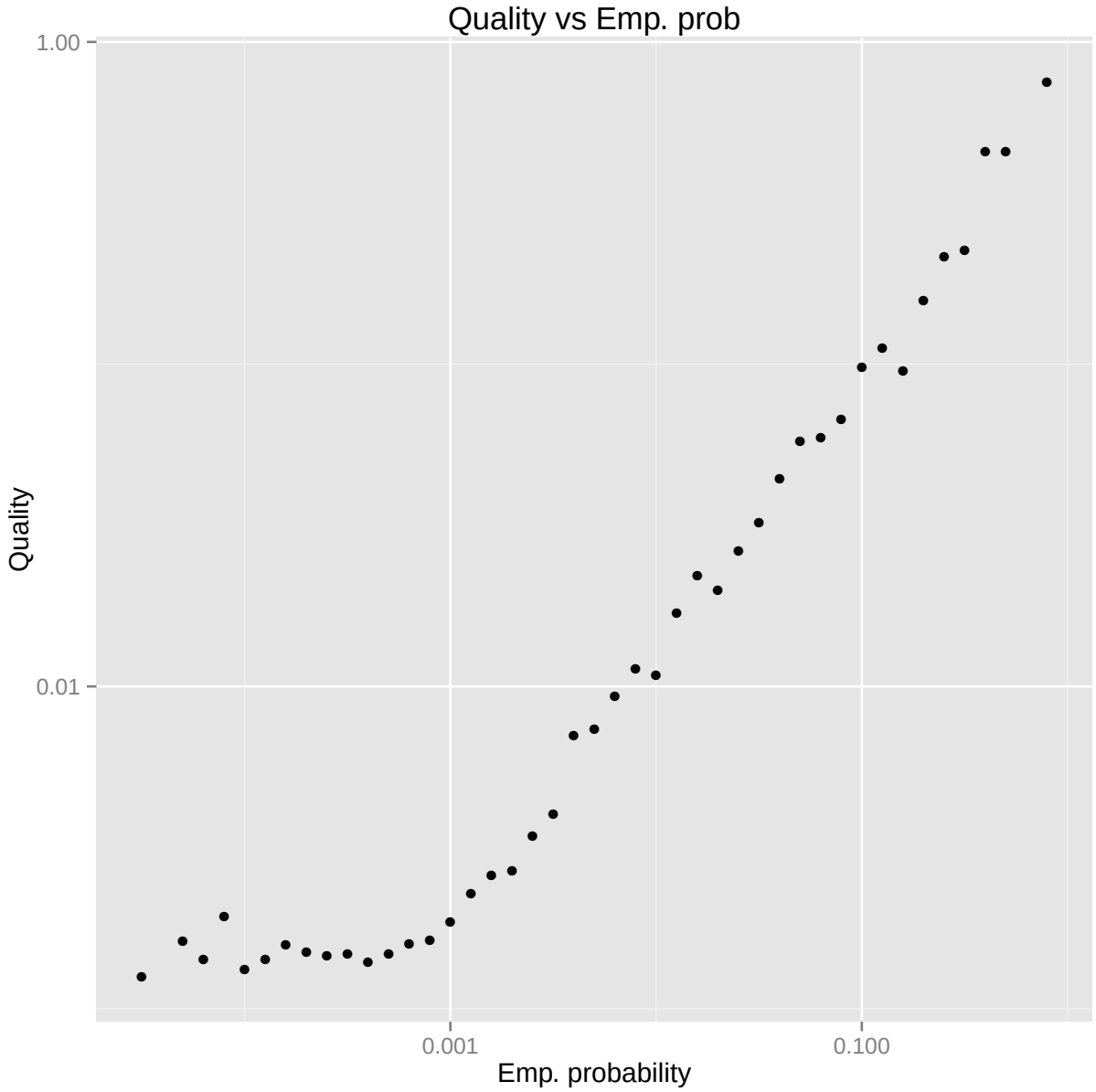

Рис. 1.3. Зависимость quality-значений от эмпирической вероятности ошибки. По оси x отложена эмпирическая вероятность ошибки, по оси y — quality-значение (в шкале $p = e^q$, где q — quality-значение). В случае, когда quality-значения являются логарифмическим преобразованием оценки вероятности ошибки ожидается, что график будет представлять из себя прямую из $(0, 0)$ в $(1, 1)$.

Глава 2

Метод исправления ошибок

Алгоритм коррекции ошибок IonHammer является обобщением метода BayesHammer [1], предназначенного для исправления ошибок вида «замена», на ошибки вида «вставка» и «удаление». IonHammer реализован на языке программирования C++ и доступен в составе геномного ассемблера SPAdes по адресу http://cab.spbu.ru/software/spades/.

2.1. Основная идея алгоритма BayesHammer

Прежде, чем переходить к описанию IonHammer, кратко опишем основные идеи алгоритма BayesHammer. Для описания алгоритма BayesHammer на потребуются следующие определения:

Определение 7. Расстоянием Хэмминга двух строк равной длины x и y из алфавита $\mathbb{A}$ равно количеству позиций, в которых данные строчки различаются.

Определение 8. ED_l-графом (англ. Edit-distance graph) для функции $d(x, y)$ называется граф, вершинами которого являются k-меры, а ребра между двумя вершинами x и y проведены тогда и только тогда, когда $d(x, y)$ меньше l.

Определение 9. Графом Хэмминга (сокращено HG_l-граф) называется ED_l-граф, в котором в качестве функции d используется расстояние Хэмминга.

Алгоритм BayesHammer основан на предположении, что для некоторого k верно:

- Информация о геноме g может быть сохранена в виде его подстрок длины k.
- Этой информации достаточно для исправления ошибок в чтениях этого генома.
- Оценить множество геномных подстрок длины k (т.е. таких подстрок, которые содержатся в геноме) можно на основе имеющихся чтений за счет того, что таких строчек много, а ошибки происходят достаточно редко.

Понятно, что для k равного длине генома первые два предположения верны, а последнее не верно. Аналогично для небольшой длины (предельный случай $k = 1$)

последнее предположение очевидно будет верно практически всегда, а первые два — нет.

На самом деле оказывается, что для большинства биологических данных все предположения будут выполнены для достаточно небольших k. В частности, алгоритмы сборки генома на основе графов де Брюина [5] даже с относительно небольшими k-мерами могут восстанавливать большие последовательные участки генома. Например, в сборщике генома SPAdes для чтений, полученных с помощью IonTorrent, используются $k = 21,\ 33,\ 55$.

Параметр k выбирается, исходя из следующих предположений:

- k должно быть достаточно мало, чтобы множество k-меров, встретившихся в чтениях, можно было сохранить в оперативной памяти компьютера.
- k должно быть достаточно большим, чтобы можно было разделить «похожие» участки ДНК. Например, для $k = 6$ участок ДНК с последовательностью нуклеотидов ATGCGCGCGCGCTGA будет сложно отличить от участка ATGCGCGCGCTGA — все возможные 6-меры для таких строк одинаковы, хотя сами строки отличаются.

BayesHammer состоит из двух основных частей:

1. Алгоритм оценки множества геномных k-меров (геномным k-мером мы называем k-мер, который встречается в геноме g).
2. Алгоритм исправления ошибок, использующий множество геномных k-меров для исправления ошибок в чтениях.

Первый шаг алгоритма BayesHammer является существенным улучшением идеи кластеризации графа Хэмминга, предложенной в [6]. В [6] на основе встретившихся в чтениях k-меров строится HG_l-граф, центры компонент связности которого объявляются геномными, а все остальные k-меры в компоненте связности предполагаются ошибочными. Такой подход к определению множества геномных k-меров имеет существенный недостаток — в геномах живых организмов часто встречаются повторяющиеся последовательности, незначительно отличающиеся друг от друга. Такие последовательности попадают в одну компоненту связности и все, кроме одной, начинают считаться

ошибками. Также при больших количестве чтений возникают ситуации, когда два геномных k-мера могут оказаться «связаны» длинной цепочкой ошибочных k-меров и также оказаться в одной компоненте связности. Такая особенность алгоритма нежелательна для большинства биологических приложений. Поэтому в BayesHammer в данный алгоритм внесена существенная модификация — дополнительная m-средних кластеризация больших компонент связности. Для технологии Illumina quality-значения для прочитанных нуклеотидов являются хорошей оценкой логарифма вероятности ошибки в данном символе. За счет этого удается определить вероятностную модель кластера и использовать BIC-критерий для оценки оптимального числа кластеров m.

Второй шаг алгоритма использует оценки геномных k-меров из первого шага для исправления ошибок в чтениях. Алгоритм предполагает, что первый k-мер в чтении не содержит ошибок. Затем он идет слева направо, исправляя ошибки в каждом следующем k-мере, если данный k-мер не оценен как геномный. В случае нескольких возможных коррекций перебираются все возможные варианты исправления и из них выбирается лучшая. Для того, чтобы избежать экспоненциального времени работы в худшем случае в алгоритме предусмотрено некоторое количество эвристик, основанных на профиле ошибок технологии Illumina.

2.2. IonHammer

Алгоритм исправления ошибок IonHammer является обобщением алгоритма BayesHammer на метод секвенирования IonTorrent. Для данной технологии естественным обобщением k-мера является hk-мер — k-мер в пространстве гомополимеров. Аналогично BayesHammer в алгоритме IonHammer можно выделить две основные компоненты:

1. Алгоритм оценки геномных hk-меров.
2. Алгоритм коррекции чтений на основе множества оценок геномных hk-меров, полученном на первом шаге.

В новой версии IonHammer мы существенно изменили первый шаг алгоритма и полностью изменили алгоритм коррекции из пункта 2.

Рис. 2.1. Блок-схема алгоритма оценки множества геномных hk-меров в IonHammer.

2.3. Алгоритм оценки множества геномных hk-меров

Блок-схема алгоритма оценки множества геномных hk-меров в IonHammer представлена на рис. 2.1. Аналогично BayesHammer мы производим кластеризацию множества hk-меров, встретившихся в чтениях и используем центры получившихся кластеров в качестве оценок геномных hk-меров. Для кластеризации мы задаем на множестве $\mathbb{H}^k \times \mathbb{H}^k$ функцию $d(x, y)$, показывающую, насколько «близки» hk-меры x и y (мы не требуем, чтобы $d(x, y)$ являлась метрикой) и строим для этой функции ED_l-граф. Получившиеся кластера подвергаются дополнительной субкластеризации для выделения геномных hk-меров, находящихся на небольшом расстоянии друг от друга. Рассмотрим подробнее все шаги алгоритма.

2.3.1. Статистики по встретившимся hk-мерам

Первые шаг алгоритма оценки множества геномных hk-меров заключается в нахождении всех hk-меров, встретившихся в чтениях и вычислении необходимых в дальнейшем статистик. Для хранения все интересующих нас данных в IonHammer строится идеальная минимальная хэш-функция. Выбор быстрого и эффективного по памяти метода построения хэш-функции для множества k-меров (и hk-меров) выходит за рамки данной работы. В текущей версии IonHammer используется метод, описанный в [7]. Во всем дальнейшем тексте мы предполагаем, что нам задана хэш-функция для всех hk-меров и мы можем получать и обновлять любые статистики для hk-мера за $O(1)$; за

такое же время мы можем проверить для произвольного hk-мера x, встречался ли он в входных данных.

В IonHammer для всех hk-меров мы будем вычислять две аддитивные статистики:

1. $C(x)$ — количество раз, которое hk-мер x встретился в чтениях.

2. $Q(x)$ — качество hk-мера x, которое определено следующим образом: пусть hk-мер x встретился в строчках $s_1, \ldots, s_m$. Не умаляя общности будем считать, что x — первый hk-мер в каждой из этих строк. Напомним, что каждой строке s сопоставлен вектор quality-статистик q_s. Обозначим соответствующие $s_1, \ldots, s_m$ quality-статистики за $q_1, \ldots, q_m$. Тогда с помощью формулы (1.1) можно вычислить $Q(s_i[0,k), q_i[0,k))$. Определим $Q(x)$ следующей формулой:

$$Q(x) = \sum_{i=1}^{m} Q\left(s_i[0,k), q_i[0,k)\right)$$

   В случае, когда q_i — логарифмы вероятностей ошибки данная формула просто задает логарифм вероятности того, что x является ошибочным. Таким образом, чем меньше $Q(x)$, тем лучше hk-мер.

   Мы проверили, что данная статистика действительно является мерой качества hk-мера. Для этого на наборе чтений с известным референсным геномом была построена зависимость вероятности ошибки в hk-мере от значения статистики $Q(x, q_x)$, где x — hk-мер, а q_x — соответствующий этому hk-меру вектор quality-статистик (статистика для hk-мера, определенная формулой (1.1)). Данная зависимость изображена на рис. 2.2. По оси x заданы значения $e^{Q(x,q_x)}$, а на оси y изображена эмпирическая вероятность ошибки в hk-мере с таким значением $e^{Q(x,q_x)}$. В случае, когда $Q(x, q_x)$ является хорошей оценкой логарифма вероятности ошибки, на графике должна быть изображена прямая линия из точки $(0, 0)$ в точку $(1, 1)$. В нашем случае это не так, но тем не менее видно, что качество hk-мера сильно коррелирует с вероятностью ошибки и может быть использована как некоторая мера качества hk-меров.

Замечание. Вычисляемые нами статистики хорошо коррелируют с вероятностью того, что hk-мер является геномным. Возникает резонный вопрос, а не достаточно ли их для того, чтобы оценить множество геномных hk-меров? Если покрытие генома чтениями равномерное, а все hk-меры в геноме уникальные (встречаются один раз), то

все относительно просто — у нас есть некоторый «средний» уровень ошибок, который для современных секвенсеров достаточно мал (менее 5%) и известен заранее (является характеристикой оборудования). В таком случае разбить hk-меры на множество геномных и негеномных не составит большого труда — покрытие hk-меров можно хорошо моделировать с помощью смеси распределений Пуассона (одна компонента — геномные hk-меры, другая ошибочные) [8]. С помощью EM-алгоритма оценим параметры распределения и с помощью формулы Байеса вычислим апостериорные вероятности того, что hk-мер геномный. Либо можно пытаться использовать EM-алгоритм для разделения пар $(C(x), Q(x))$, подобрав хорошее совместное распределение для $(C(x), Q(x))$. К сожалению, такой подход нам не подходит — в биологических данных нельзя ожидать того, что все hk-меры в геноме уникальны, а покрытие будет равномерным (для таких k, для которых мы можем собрать достаточно статистики). Более того, как было замечено в анализе профиля ошибок технологии IonTorrent, чем больше длина гомополимера, тем больше ошибок в нем происходит. В результате уровень ошибок в hk-мерах, содержащих несколько длинных гомополимеров существенно выше, чем средний уровень ошибок по всем чтениям. Таким образом, одной простой моделью обойтись не удается и требуются более сложные подходы. Кроме того, естественно ожидать, что подход, учитывающий больше информации, будет работать лучше подхода, который эту информацию никак не использует.

2.3.2. Кластеризация ED_l-графа

Первый шаг алгоритма, находящий компоненты связности ED_l-графа, предназначен для разбиения всех hk-меров на относительно небольшие множества, в которых содержится небольшое количество геномных hk-меров. Для этого требуется такая функция $d(x, y)$, что

1. Для большой части ошибочных hk-меров выполнено: если y получен в результате чтения hk-мера x, то $d(x, y) \leq l$.

2. Для большой части пар геномных hk-меров выполнено: если x и y два различных геномных hk-мера, то $d(x, y) > cl$, где $c \geq 1$ некоторая константа (которую мы напрямую контролировать не можем, зависит от функции d и генома). Чем больше c, тем меньше близких геномных hk-меров будут попадать в одну компоненту

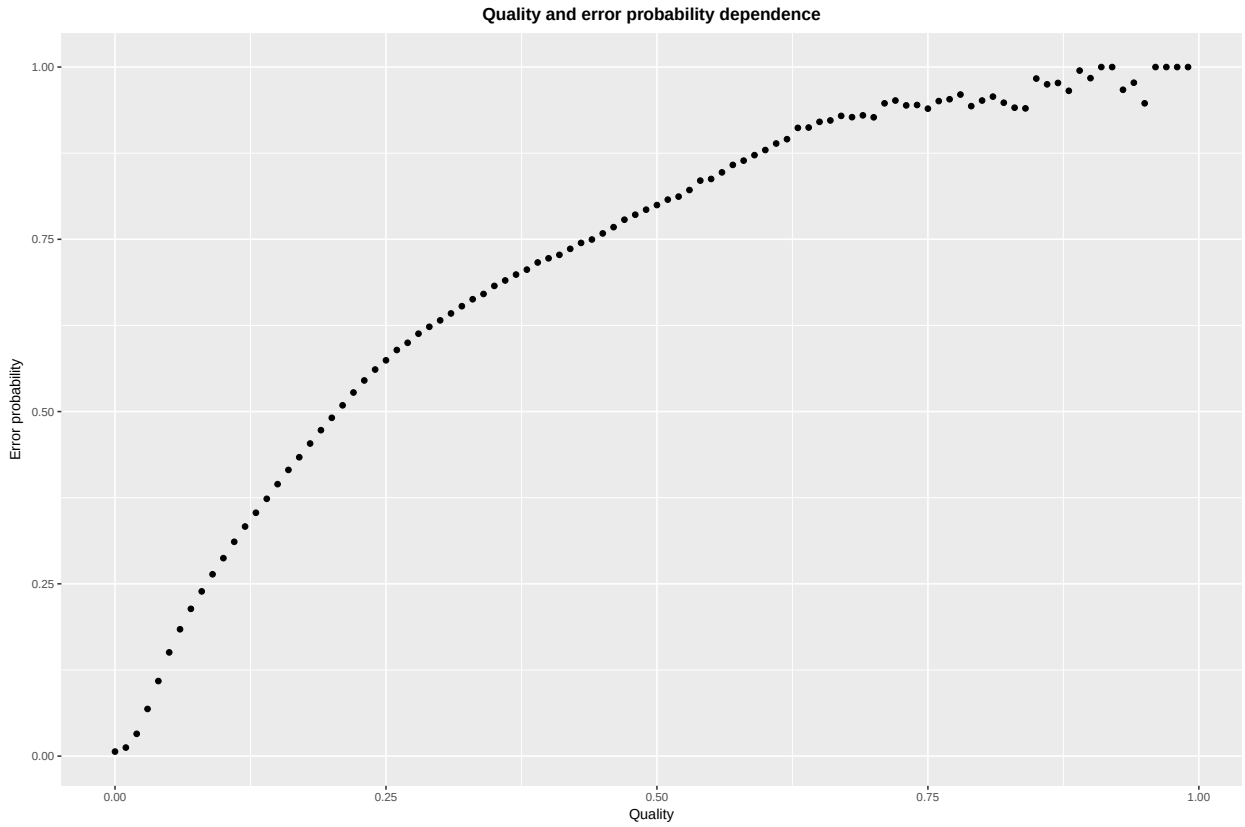

Рис. 2.2. Зависимость вероятности ошибки от значения quality-статистики.

связности.

При выборе функции d и параметра l, а также размера hk-меров k стоит учитывать, что для применения алгоритма в практических задачах необходимо, чтобы можно было эффективно искать компонент связность ED_l-графа. Параметры d, l, k зависят друг от друга и их выбор существенно влияет на то, как стоит реализовывать поиск компонент связности. На основе изучения профиля ошибок технологии IonTorrent и экспериментов на нескольких наборов чтений бактерий мы пришли к выводу, что для IonHammer достаточно использовать hk-меры с $k = 16$. Для таких hk-меров в качестве расстояния d оказалось возможным, как показали проведенные нами эксперименты, использовать обобщение расстояние Хэмминга на пространство hk-меров (определение 10).

Определение 10. Расстояние Хэмминга двух строк x и y из алфавита $\mathbb{H}$ определим следующим образом:

- Если существует позиция, в которой нуклеотиды не совпадают, то считаем расстояние между x и y равным бесконечности.

- В противном случае расстояние равно $\sum_{i=1}^{k} |\text{len}(x_i) - \text{len}(y_i)|$ — абсолютной разнице длин гомополимеров на одинаковых позициях.

Для данного расстояния на рис. 2.3 изображено количество ошибочных hk-меров в зависимости от их расстояния до соответствующего геномного hk-мера. Из рисунка видно, что большая часть ошибочных hk-меров находится на расстоянии 1. В результате мы решили, что в IonHammer будет всегда использоваться следующий набор параметров:

1. $k = 16$.
2. $d(x, y)$ — расстояние Хэмминга.
3. $l = 1$.

Такой выбор параметром позволяет реализовать эффективный алгоритм построение компонент связности ED_1-графа. Для этого будем использовать структуру данных «Система непересекающихся множеств» [9]. Данная структура позволяет проводить следующие операции:

- Для элементов x и y объединить множество, содержащее элемент x и множество, содержащее элемент y.
- По элементу x найти, в каком множестве находится данный элемент.

Существует реализация данной структуры данных, в которой для любых практических приложений можно считать, что операции выполняются в среднем за $O(1)$.

Для кластериазции полноценный ED_1-граф не требуется и достаточно для каждого hk-мера x найти множество hk-меров, которые находятся с ним в одной компоненте связности. Для этого для каждого hk-мера x можно сгенерировать не более $2k$ hk-меров, которые могут находится на расстоянии 1 от него. Если сгенерированный hk-мер встречался в чтениях, то мы объединим множества, содержащие данные hk-меры в одно (изначально каждый элемент содержится в множестве размера один, состоящем только из данного hk-мера). Проверка на то, встречался ли hk-мер в входных данных можно производить за $O(1)$ при использовании хэш-таблицы. Таким образом, для всего алгоритма кластеризации потребуется $O(kN)$ операций, где N — количество hk-меров, встретившихся в чтениях. Важно отметить, что возможна реализация системы непересекающихся множеств, с которой можно эффективно работать в многопоточных программах. За

счет этого алгоритм поиска компонент связности можно выполнять параллельно и эффективно задействовать все логически ядра процессора, доступный на компьютере.

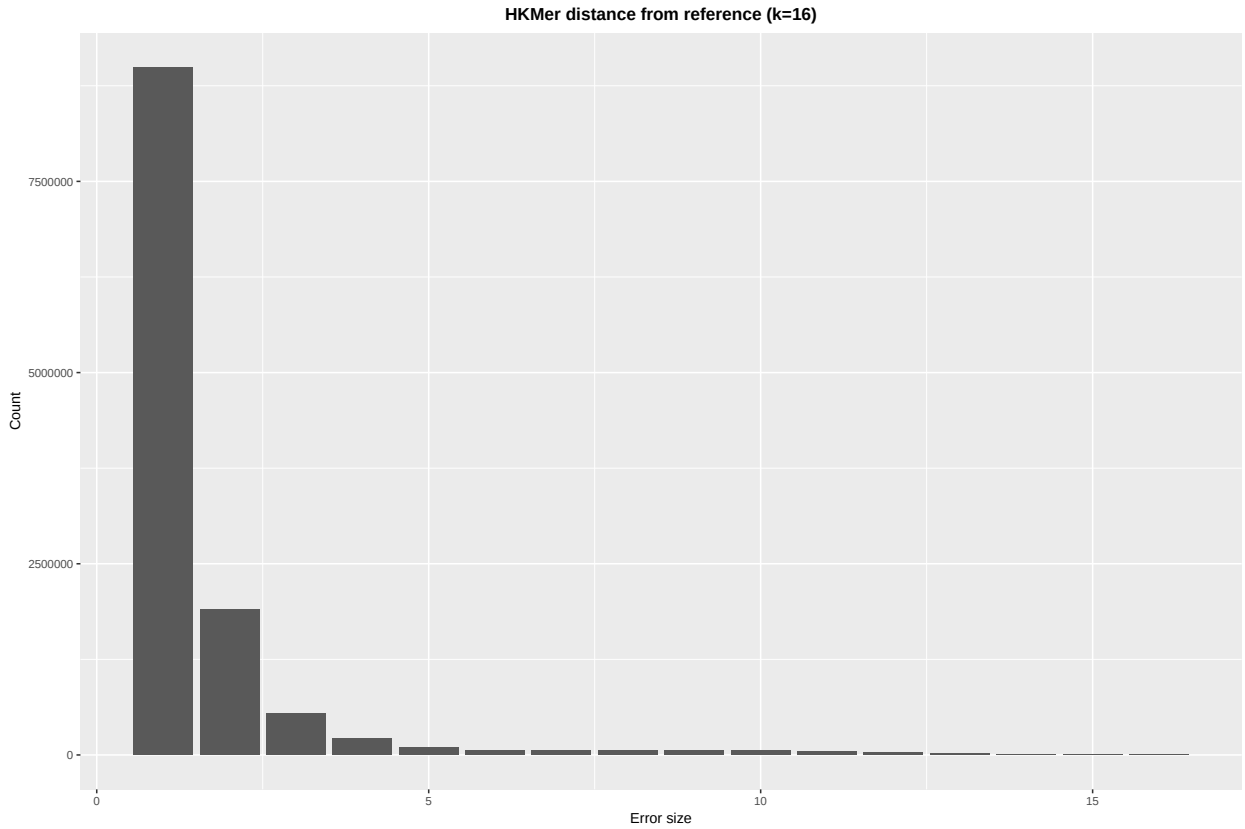

Рис. 2.3. Количество ошибочных hk-меров x в зависимости от расстояния до референсного (правильного) hk-мера x_c.

2.3.3. Субкластеризация

В результате первого шага алгоритма мы получаем множество компонент связности ED_1-графа. Получившиеся компоненты можно разделить на 3 группы:

1. Компоненты, содержащие один геномный hk-мер и ошибочные, полученные из него.
2. Компоненты связности, содержащие несколько геномных hk-меров.
3. Компоненты, образованные полностью ошибочными hk-мерами.

Нам требуются два алгоритма. Во-первых, необходимо разбить компоненты с несколькими геномными hk-мерами на более маленькие — данный процесс мы назы-

ваем субкластеризацией. Во-вторых, необходимо научиться фильтровать компоненты третьего типа.

Для того, чтобы разбить большие компоненты связности на более маленькие кластера мы, как и в BayesHammer, используем кластеризацию с помощью метода m-средних. Для выбора параметра m вместо BIC-критерия используется более простой набор эвристик, т.к. значения quality-статистик для данных, полученных с помощью IonTorrent, трактовать как вероятности нельзя и написать хорошую вероятностную модель не удается.

Субкластеризация и фильтрация ошибочных центров основана на следующей идее: для каждого hk-мера x мы считаем статистики $Q(x)$ и $C(x)$. Чем больше $C(x)$ и меньше $Q(x)$, тем больше шансов на то, что данный hk-мер геномный. Использовать абсолютные значения $C(x)$ и $Q(x)$ достаточно сложно — из-за того, что покрытие генома неравномерное абсолютные статистики ошибочных hk-меров из регионов с большим покрытием по вполне естественным причинам могут быть неотличимы от статистик геномных hk-меров из регионов с средним или маленьким покрытием. Изучая эмпирические данные мы обнаружили, что в качестве меры качества можно рассматривать статистику вида $\frac{Q(x)}{C(x)+\lambda}$. На рис. 2.4 изображены эмпирические плотности такой статистики, зеленым цветом выделена компонента, относящаяся к геномным hk-мерами, а красным цветом компонента, относящаяся к ошибочным hk-мерам. Из рисунка видно, что компоненты хорошо разделяются и могут быть использованы для фильтрации хороших hk-меров от плохих. Для этого мы моделируем распределения компонент с помощью смеси нормальных распределений (для негеномной компоненты можно также брать и несимметричное распределение, как видно из рисунка, но и самый простой вариант показал хорошие результаты). Для оценки параметров используется EM-алгоритм. Далее в качестве хороших hk-меров берутся такие hk-меры, для которых апостериорная вероятность того, что данный hk-мер из левой (геномной) компоненты больше некоторого порога, в качестве которого в IonHammer используется 0.5.

К сожалению, такая фильтрация не подходит для определения числа геномных hk-меров в компоненте связности. Ошибочные hk-меры, находящиеся на расстояние один от геномных, на основе средней статистики разделяются гораздо хуже. Из-за этого на шаге субкластеризации мы используем более консервативный подход к определению числа субкластеров m, включающий в себя ряд эвристик. Во-первых, в качестве

порогового значения мы берем не 0.5, а число, близкое к 1 (в IonHammer по-умолчанию используется 0.999). При определении числа субкластеров мы не рассматриваем в качестве потенциальных центров такие hk-меры, на расстоянии один от которых существуют hk-меры, встретившиеся существенно большее число раз.

Обобщая вышесказанное, в качестве потенциальных центров кластеров мы рассматриваем hk-меры, удовлетворяющие следующим условиям:

- Апостериорная вероятность того, что hk-мер геномный больше порогового значения α
- Для кандидата на геномный hk-мер x не существует такого hk-мера y, что $\beta C(y) > C(x)$, где β некоторый параметр, зависящий от hk-мера y. В IonHammer в качестве β используется консервативная верхняя граница на частоту ошибок, возникающих при чтении hk-меров длины $|y|$, оцененная на основе эмпирических данных. Отметим, что эта эвристика необходима. В данных встречаются hk-меры на расстоянии 1 от геномных, которые по апостериорной вероятности отделить от геномных нельзя.

В качестве m мы берем максимум из единицы и числа hk-меров, удовлетворяющих данным условиям.

2.3.4. Отличия от предыдущей версии IonHammer

В предыдущей версии IonHammer для оценки геномных центров также использовалась кластеризация ED_l-графа. Но, в отличии от нового алгоритма, в качестве функции d использовалось расстояние Левенштейна, а не расстояние Хэмминга на hk-мерах. В результате этого нельзя было адаптировать эффективный метод построения компонент связности ED_1-графа для кластеризации множества hk-меров. Вместо этого hk-меры группировались на основе первой и второй половин (т.е. для hk-мера x $h\frac{k}{2}$-меры $x[0 : \frac{k}{2})$ и $x[\frac{k}{2} : k)$ использовались, как ключи для группировки). В каждой группе для каждого hk-мера x выполнялся линейный поиск всех hk-меров, находящихся на расстоянии не более l от x. После этого для всех таких hk-меров y компонента связности, содержащие hk-мер x и компонента связности, содержащая hk-мер y объединялись. Поиск в группе hk-меров требует $O(m^2)$ операций, где m — размер группы. На больших наборах данных данный алгоритм мог работать долго из-за большого чис-

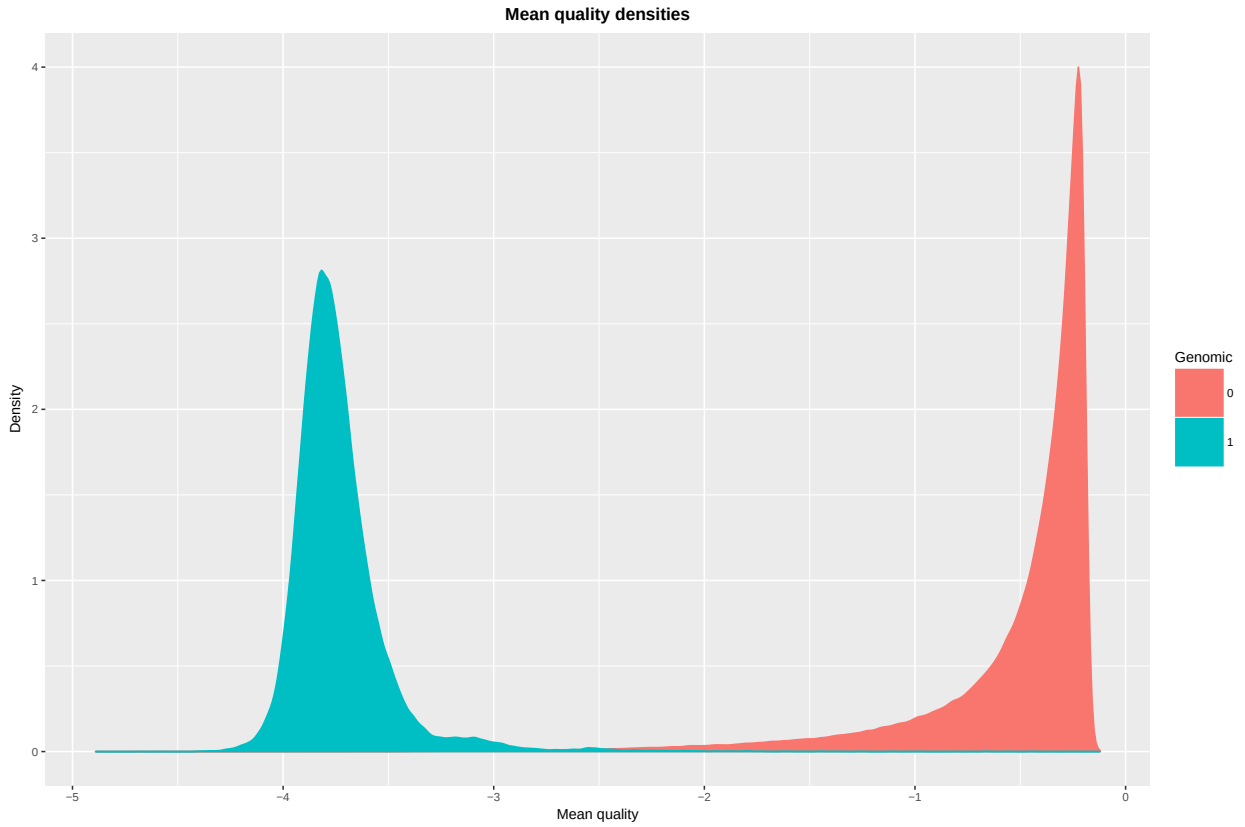

Рис. 2.4. Статистика $\frac{Q(x)}{C(x)+\lambda}$ для центров компонент связности ED_1-графа. Красным цветом выделены ошибочных hk-меры.

ла ошибочных hk-меров (при большом количестве ошибочных hk-меров увеличивается размер групп). Например, на наборе данных из 22×10^6 чтений старый алгоритм поиска компонент связности работал 27 минут (в 32 потока). Новый алгоритм кластеризации работает всего 5 минут.

Также в предыдущей версии IonHammer пороговые значения для шага субкластеризации и фильтрации требовалось задавать вручную, а в новой версии предложен автоматический метод оценки. Более того, из-за использования расстояния Левенштейна компоненты связности ED_1-алгоритма получались большими и содержали большое количество различных геномных hk-меров, из-за чего качество шага субкластеризации ухудшалось.

2.3.5. Качество алгоритма кластеризации

Для анализа качества работы алгоритма оценки множества геномных hk-меров мы посчитали на доступных нам данных следующие статистики:

- Количество центров геномных кластеров, оцененных геномными. Должно быть близко к общему числу геномных hk-меров в чтениях.
- Количество негеномных hk-меров, оцененных геномными. Должно быть как можно меньше.
- Количество геномных hk-меров, которые были оценены как негеномные. Таких hk-меров также должно быть как можно меньше.

На всех наборах данных алгоритм оценки множества геномных hk-меров показал хорошие результаты. Результаты на нескольких наборах данных представлены в таблице 2.1.

Замечание. Стоит отметить, что кластеризация графа действительно помогает уменьшить количество ошибочных hk-меров. Мы проверили, что будет, если применить метод фильтрации (предложенный в разделе 2.3.3) не для центров компонент связности, а для всех hk-меров. Оказалось, что тогда количество ошибочных hk-меров, оцененных геномными, увеличивается более, чем в 10 раз.

Организм	*E. coli str. DH10B*	*E. coli str. DH10B*	*E. coli str. DH10B*	*E. coli str. O157H Sakai*
Чип	520	530	314	318
Чтений	7247730	22749163	494921	6500837
Геномные, оценные геномными	6562088 (99.78%)	6550682 (99.6%)	6492056 (98.7%)	7660600 (99.8%)
Геномные, оцененные негеномными	14325 (0.22%)	25861 (0.4%)	82806 (1.3%)	9374 (0.2%)
Негеномные, оцененные геномным	11971	39002	30131	143061
Всего hk-меров	206560123	403698029	24939947	149099955

Таблица 2.1. Количество геномных и негеномных центров. Для геномных центров указана доля об общего числа геномных hk-меров в чтениях.

2.4. Алгоритм коррекции ошибок

2.4.1. Алгоритм коррекции

Пусть дано чтение $r \in \mathcal{H}^*$. Для исправления ошибок в чтении r нам требуется уметь сравнивать две коррекции $r_1 \in \mathcal{H}^*$ и $r_2 \in \mathcal{H}^*$. Мы будем предполагать, что для

сравнения двух коррекции достаточно ввести функцию штрафа $f_r(r^*)$, для которой исправление r_1 лучше, чем исправление r_2 тогда и только тогда, когда $f_r(r_1) < f_r(r_2)$. В таком случае оптимальная коррекция чтения r может быть найдена как решение следующей задачи минимизации:

$$r^* = \arg\min_{r^* \in \mathcal{H}^*} f_r(r^*) \tag{2.1}$$

Разумеется, для произвольной функции f_r решить такую задачу эффективно невозможно и требуются дополнительные предположения о функции f_r. В IonHammer предложен алгоритм поиска решения задачи (2.1) для функций ошибок вида:

$$h : \mathcal{H}^k \times \mathcal{H}^k \to \mathbb{R}_{\geq 0} \tag{2.2}$$

$$f_r(r^*) = h(r[0,k), r^*[0,k)) + \ldots + h(r[|r|-k, |r|), r^*[|r|-k, |r|)), \tag{2.3}$$

где h — некоторый штраф за исправление одного hk-мера на другой. При этом в записи (2.3) мы будем предполагать, что последовательность нуклеотидов в r и r^* одинаковы, но допускаем, что у гомополимеров бывает длина ноль — за счет такого предположения мы можем рассматривать все ошибки как неверную оценку длины и игнорировать ошибки замены одного нуклеотида на другой.

Рассмотренный далее алгоритм является обобщением метода исправления ошибок из BayesHammer на случай ошибок вида «вставка» и «удаление». При описании алгоритма коррекции мы будем предполагать, что известно, какие hk-меры являются геномными, а какие ошибочными. На реальных данных множество геномных hk-меров не известно и вместо него мы будем использовать оценку, полученную методом, описанным в разделе 2.3. Таким образом, геномный hk-мер в контексте данного алгоритма является, на самом деле, оценкой геномного hk-мера.

Предположим, что в чтении есть хотя бы один hk-мер, измеренный без ошибок. Не умаляя общности будем считать, что это первый hk-мер. Если для чтения r это не так и геномный hk-мер начинается на позиции $i \neq 0$, то сделаем следующее:

$$r_l = r[0 : (i+k))$$
$$r_r = RC(r[i : |r|)),$$

где $RC(s)$ операция перевода последовательности нуклеотидов s в обратно-комплементарную. У r_l и r_r первый hk-мер является геномным и можно исправлять ошибки в этих

подстроках независимо. Результаты исправления r_l и r_r аналогичным образом преобразуются обратно в исправление чтения r.

Основная идея алгоритма — идти по чтению слева направо и каждый раз пытаться исправлять длину последнего гомополимера в текущем hk-мере. При этом исправление длины до нуля означает полное удаление гомополимера, а добавление на последнюю позицию нового гопомолимера трактуется как исправление в гомополимере длины ноль. Для простоты при описании алгоритма мы будем считать, что все коррекции являются изменениями длины гомополимера $(c, l), l > 0$ на гомополимер $(c, l'), l' > 0$ — т.е. не будем рассматривать полное удаление гомополимера и вставку нового гомополимера.

Для того, чтобы избежать экспоненциального роста числа возможных коррекций, алгоритм будет рассматривать только такие исправления, которые приводят к геномным hk-мерам. Соответственно алгоритм будет искать только аппроксимацию решения (2.1).

Пусть требуется исправить чтение r. Вид функции f (2.3) позволяет эффективно отсекать плохие варианты коррекции. Заметим, что штраф за исправление чтения r для любого $i \in 1 \dots |r|$ можно представить как сумму штрафа за исправление префикса $r[0 : i)$ и суффикса $r[i, |r|)$. Введем множество $\mathbb{Q}$, состоящее из троек (s, i, p), где s — некоторая коррекция $r[0 : i)$ (т.е. префикса r из первых i гомополимеров), $i \in \mathbb{N}$ — количество исправленных гомополимеров, $p \in \mathbb{R}_{\geq 0}$ — штраф за произведенные в s коррекции. На первом шаге алгоритма добавим в множество $\mathbb{Q}$ тройку $(r[0 : k), k, 0)$. Далее до тех пор, пока для тройки из $\mathbb{Q}$ с наименьшим штрафом не будет выполнено $i = |r|$ будем повторять следующую операцию:

1. Удаляем из $\mathbb{Q}$ тройку (s, i, p) с наименьшим штрафом.

2. Рассматриваем все возможные коррекции длины гомополимера $r[i]$. Для каждой коррекции x гомополимера $r[i]$, приводящей к геномному hk-меру, добавляем в $\mathbb{Q}$ тройку $(s \mid x, i{+}1, p{+}h\left((s \mid r[i])[i-k, i), (s \mid x)[i-k, i)\right)$ — т.е. добавляем все возможные коррекции длины последнего гомополимера с соответствующим обновлением штрафа за произведенную коррекцию.

Таким образом, алгоритм будет искать только среди тех коррекций, который потенциально могут быть оптимальными с точки зрения штрафа f. После завершения работы алгоритма находится оптимальное решение (при условии, что мы не рассматриваем

коррекции, приводящие к негеномным hk-мерам) — за счет того, что функция h всегда положительна, любая другая коррекция будет иметь по крайне мере такой же штраф, как найденное решение. Псевдокод алгоритма представлен в приложении А в листинге 2.

Предложенному алгоритму в худшем случае может потребоваться перебрать экспоненциальное число коррекций для завершения работы (каждая ошибка может давать несколько потенциальных исправлений). Чтобы избежать такой ситуации мы дополнительно используем следующие эвристики:

1. IonHammer не пытается исправлять hk-меры, являющиеся геномными.

2. Ограничена максимальная возможная коррекция длины — у IonTorrent большая часть ошибок имеет размер 1 и можно рассматривать только такие исправления. В IonHammer по-умолчанию рассматриваются коррекции, размер которых не превосходит 3.

3. Введено ограничения на:

   - Максимальное возможное количество исправлений в чтении.
   - Максимально возможное количество исправлений в последнем hk-мере.
   - Размер множества $\mathbb{Q}$ ограничен некоторой константой, зависящей от длины чтения — в случае, если размер множества $\mathbb{Q}$ становится больше данной константы начинаем использовать жадный алгоритм коррекции (т.е. алгоритм, который каждый раз при добавлении в $\mathbb{Q}$ рассматривает только лучшую коррекцию, а не все возможные). Важно отметить, что ситуации, когда размер множества $\mathbb{Q}$ достигает максимального, происходят крайне редко. Если множество геномных hk-меров оценено хорошо, то вариантов исправить чтение достаточно мало и переход на жадный алгоритм не происходит. Например, на всех изученных нами чтениях бактерий *E. coli str. DH10B* и *E. coli str. O157H Sakai* перехода на жадный алгоритм не происходило.

     Основное назначение данной эвристики — гарантия завершения IonHammer за разумное время на любых входных данных. При этом, если переход на жадный алгоритм происходит достаточно часто, то мы выводим предупреждение, т.к. это событие является индикатором того, что какое-то из предположений, лежащих в основе IonHammer нарушено.

Замечание. Для простоты описания алгоритма мы предположили, что среди коррекций нет полного удаления гомополимера, а также вставки нового гомополимера. Такие ошибки также встречаются и в IonHammer описанный подход исправления ошибок обобщен на полные вставки и удаления гомополимеров. Не будем вдаваться в технические подробности и опишем основную идею того, как обрабатываются ошибки такого рода.

Будем считать, что вставка нового гомополимера (c, l) — исправление гомополимера $(c, 0)$ на гомополимер (c, l). Аналогично удаление (c, l) это исправление данного гомополимера на $(c, 0)$.

Функция штрафа (2.3) допускает длины гомополимеров, равные нулю, так что вычислить штраф не составляет труда.

В случае, когда требуется удалить гомополимер на i-ой позиции мы просто считаем, что длина гомополимера равна нулю. Остается понять, что считать исправлением, приводящим к геномному hk-меру — мы считаем, что коррекция приводит к геномному hk-меру, если после удаления гомополимера следующий hk-мер является геномным. Пусть дано чтение r, в котором $r[0, k)$ геномный, а $r[1, k]$ негеномный. Тогда удаление гомополимера $r[k]$ мы рассматриваем только в случае, когда $r[1, k) \,|\, r[k + 1]$ будет геномным.

Для ситуации, когда требуется вставить новый гомополимер (c, l), мы предполагаем, что на i-ой позиции на самом деле записан гомополимер длины 0, т.е. преобразуем чтение r в чтение $r[0 : i) \,|\, (c, 0) \,|\, r[i : |r|)$. Исправление i-ого гомополимера $(c, 0)$ на (c, l) приводит к геномному hk-меру, если $r[0 : i) \,|\, (c, l)$ является геномным. При этом мы не допускаем вставок несколько раз подряд, т.к. иначе можно будет «сгенерировать» любой геномный hk-мер.

2.4.2. Функция штрафа

Качество работы алгоритма существенным образом зависит от определения функции штрафа h. При выборе данной функции для IonHammer мы исходили из следующих предположений:

- Исправленное чтение должны находиться «близко» к исправляемому чтению.
- В чтении должно быть как можно меньше оцененных негеномными hk-меров.

- При прочих равных надо выбирать такую коррекцию, которая обеспечивает наименьшее отличие в значениях статистики $C(x)$ (количество раз, которое встретился hk-мер) у соседних hk-меров.

В итоге мы остановились на функции вида:

$$h(x, x_c) = \alpha d(x, x_c) + \beta O(x_c) + \gamma \log P(C(x_c)|\text{read}), \tag{2.4}$$

где

1. $d(x, y)$ — расстояние Хэмминга между двумя hk-мерами.
2. $O(x)$ — функция-индикатор, равная 1, если x оценен как негеномный hk-мер и нулю иначе.
3. $P(C(x)|\text{read})$ — «вероятность» встретить статистику $C(x)$ среди оцененных геномными hk-меров в исправляемом чтении. Подробное обсуждение данной компоненты будет представлено дальше.
4. α, β, γ — подбираемые вещественные параметры. Из вида функции штрафа следует, что $\alpha, \beta \geq 0$, $\gamma \leq 0$. Для IonHammer были подобраны α, β, γ, дававшие (в некотором смысле) оптимальное качество на нескольких наборах реальных данных.

Первые две компоненты особого обсуждения не требуют — мы вводим штраф за любую коррекцию hk-мера в чтении, линейной-зависящий от расстояния Хэмминга, на котором оказывается исправленный hk-мер от наблюдаемого, а также константный штраф за каждый негеномный hk-мер, встретившийся в чтении.

Цель третье компоненты — выбрать правильную коррекцию в случае, когда есть несколько хороших альтернатив (например, в случае, когда негеномный hk-мер был ошибочно оценен геномным). В рамках одного чтения можно предположить, что покрытие близко к равномерному и можно ожидать, что статистики $C(x)$ не будут «сильно» отличаться друг от друга для всех геномных hk-меров в чтении. В действительности, это не совсем верно из-за того, что некоторые hk-меры могут повторяться в разных частях генома, но, тем не менее, такое предположение достаточно часто не сильно далеко от истины. В частности, алгоритм коррекции Pollux [10] исходит из схожих предположений.

В каждом чтении у нас есть набор hk-меров, которые мы оценили как геномные. Пусть это hk-меры $x_1, x_2, \ldots, x_m$. Выберем некоторое параметрическое семейство распределений $\{P_\theta\}$ и предположим, что $\forall i\ C(x_i) \sim P_\theta$ для некоторого θ. Теперь с помощью метода максимального правдоподобия оценим параметр θ по статистикам $C(x_1), \ldots, C(x_m)$. Конечно, сам метод применять не очень корректно — у нас данные зависимы. Но наша цель — подобрать некоторую функцию «похожести», а не настоящее распределение и для таких целей все действия корректны. Далее будем использовать функцию $\log(P_\theta)$ при вычислении штрафа за коррекции — данный штраф автоматически будет отдавать при исправлении hk-мера x на x_c предпочтения таким коррекциям, которые делают статистику $C(x_c)$ «близкой» к $C(x_1), \ldots, C(x_m)$.

Выбор параметрического семейства $\{P_\theta\}$ может быть, в принципе, любым — мы просто ищем функцию штрафа. В IonHammer было решено использовать гамма-пуассоновское распределение (определение 11). Мотивация к выбору следующая — при равномерном покрытии для всех k-меров x статистики $C(x)$ хорошо аппроксимируется распределением Пуассона (с общим параметром λ). В нашем случае не все hk-меры равнозначны (в длинных hk-мерах ошибки встречаются чаще), поэтому стоит ожидать, что один параметр λ для всех hk-меров не подойдет. Гамма-пуассоновское распределение можно трактовать как распределение Пуассона с априорным гамма-распределением на параметре интенсивности λ, либо как более робастную версию распределения Пуассона.

Для гамма-пуассоновского распределения оценка параметров с помощью метода максимального правдоподобия в явной форме отсутствует и для оценки параметров в IonHammer используется метод Ньютона.

Определение 11. Гамма-пуассоновским распределением с параметрами α, β называется распределение случайной величины ξ, полученной по следующему алгоритму:

$$\lambda \sim Gamma(\alpha, \beta),$$

$$\xi \sim Poisson(\lambda).$$

Это распределение также известно под названием отрицательное биномиальное распределение (которое, обычно, записывается в другой параметризации).

Глава 3

Оценка качества алгоритма

Для оценки качества работы IonHammer мы взяли наборы чтений для бактерии *E. coli DH10B* (список представлен в таблице Б.1) и чтений для бактерии *E. coli 0157H Sakai* (список чтений в таблице Б.2). Все наборы данных доступны с сайта ThermoFisher (производитель оборудования IonTorrent) по адресу https://www.thermofisher.com/datasets/. Все эксперименты проводились на сервере с двумя Intel E5-2650 CPU (8 физических ядер, 16 логических) и 256GB оперативной памяти.

3.1. Сравнение новой версии IonHammer со старой

Первая задача, которая стояла перед нами — сравнить новый алгоритм коррекции в IonHammer со старым. Для сравнения мы использовали несколько наборов чтений, для которых параметры по-умолчанию в старой версии IonHammer показывали близкие к оптимальным результаты. Кроме того, результаты оценки геномных центров для этих наборов данных практически не отличались в новой и старой версии IonHammer, за счет чего мы могли сравнить качество работы алгоритмов коррекции без влияния метода оценки центров. Новая версия IonHammer исправляла в несколько раз большее число ошибок, а также работала существенно быстрее. Одна из основных идей в IonHammer — замена плохих hk-меров на оценки геномных центров. Поэтому основной метрикой, по которой мы сравнивали старый и новый алгоритм было то, насколько много геномных hk-меров (должно быть близко к общему число геномных hk-меров в чтениях) и негеномных hk-меров (должно быть как можно меньше) будет в исправленных чтениях. Кроме того, мы также вычислили следующие статистики:

1. Количество чтений, выравненных на ту же часть генома после коррекции, что и до — чем больше значение, тем лучше;

2. Количество «испорченных» чтений — чтения, оказавшиеся после коррекции на большем расстоянии от референсной строки, чем до — чем меньше значение статистики, тем лучше;

3. Количество полностью исправленных чтений — чем больше, тем лучше;
4. Количество полностью и частично исправленных чтений — чем больше, тем лучше;
5. Количество исправленных чтений с одной ошибкой — чем больше, тем лучше;
6. Количество испорченных чтений с одной ошибкой — чем меньше, тем лучше;
7. Среднее расстояние Левенштейна после коррекции до референсной строки — чем меньше, тем лучше;
8. Количество чтений, выравненных на другую геномную позицию после коррекции – чем меньше, тем лучше;

Результаты сравнения с наиболее важными метриками представлены в таблице 3.1 и таблице 3.2. Полные таблицы доступные в приложении. В скобках для количества геномных hk-меров указана доля от общего числа hk-меров в геноме; для количества негеномных hk-меров — во сколько раз негомных hk-меров было больше до коррекции; для расстояния Левенштейна — относительное улучшение по сравнению с расстоянием, которое было до коррекции.

	Новый	Старый
Время работы (мин., 32 потока)	2:40	5:02
Кол-во геномных hk-меров	5990612 (91.0893%)	5990664 (91.0901%)
Кол-во негеномных hk-меров	2402470 (x6.5)	6355675 (x2.45)
Среднее расстояние Левенштейна	2.1570 (x2.28)	3.5669 (x1.38)
Испорченные и выравненные на другую геномную позицию	29977 (4.62%)	54757 (8.44%)

Таблица 3.1. Сравнение нового и старого алгоритма. Набор чтений SN2-158, секвенирован с помощью чипа 314. Всего чтений 649414, 96.4% содержат хотя бы одну ошибку, средняя длина 384. Всего в чтениях 21586815 hk-меров, из них геномных 5991254 (91.0991% от общего числа hk-меров в геноме)

	Новый	Старый
Время работы (мин., 32 потока)	1:22	2:55
Кол-во геномных hk-меров	6553831 (99.65%)	6567815 (99.86%)
Кол-во негеномных hk-меров	716331(x14)	3661340 (x2.7)
Среднее расстояние Левенштейна	0.2542 (x4.86)	0.6755 (x1.83)
Испорченные и выравненные на другую геномную позицию	33676 (4.92%)	40037 (5.86%)

Таблица 3.2. Сравнение нового и старого алгоритма. Набор чтений C24-698. Всего чтений 685406, среди них 338748 (49.4%) содержат ошибки. Чтения секвенированы с помощью 314 чипа. Всего в чтениях 16779228 hk-меров, среди которых геномных 6576115 (99.9921% от общего числа hk-меров в геноме).

В результате мы выяснили, что новая версия IonHammer исправляет существенно большее число ошибок, количество ошибочных hk-меров в несколько раз меньше по сравнению с предыдущей версией, отличия в количество геномных hk-меров не существенны. Кроме того, новая версия «портит» меньшее число чтений, что также является важным показателем качества работы алгоритма. В итоге мы пришли к выводу, что новый метод коррекции работает лучше и в дальнейших исследованиях использовали только новую версию IonHammer.

3.2. Сравнение IonHammer с другими алгоритмами коррекции

Как мы выяснили в прошлом разделе, новая версия IonHammer работает лучше, чем старая. Далее перед нами стояла задача сравнить новую версию IonHammer с другими алгоритмами коррекции, позволяющими исправлять ошибки вида «вставки и удаления». Среди алгоритмов, показывающих наиболее высокое качество коррекции можно выделить Pollux [10], Coral [11] и Fiona [12].

При сравнении алгоритмов нас интересовало несколько вещей. Во-первых, от алгоритма коррекции требуется, чтобы он исправлял ошибки в чтениях. Для оценки качества работы алгоритмов мы для всех наборов данных вычислили метрики, описанные в предыдущем разделе.

Во-вторых, от алгоритмов требуется высокая скорость работы и разумный расход оперативной памяти компьютера — алгоритм, которому для исправления ошибок требуется несколько дней работы использовать достаточно сложно, даже если он хорошо исправляет ошибки. Мы измерили скорость работы и расход памяти всеми алгоритмами на нескольких наборах данных, результаты для двух наборов чтений представлены в приложении.

В-третьих, достаточно часто исправление ошибок является предварительным шагом для дальнейшей обработки чтений и решения различных биологических задач. Одной из важных биологических задач является сборка генома на основе имеющихся чтений. При этом время сборки генома существенно зависит от того, насколько много ошибок встречается в входных данных. Поэтому, если алгоритм коррекции может быстро исправить большую часть ошибок, то такой алгоритм может как ускорить время сборки генома, так и улучшить качество собранного генома. Некоторые алгоритмы исправления ошибок могут допускать систематически ошибки, что негативно влияет на результаты сборки генома и в таком случае даже быстрый метод коррекции использовать не получится. Поэтому мы для всех наборов данных проверили также то, как влияет исправление ошибок на время и качество работы геномного ассемблера SPAdes (использовалась версия SPAdes 3.10.1).

На всех наборах данных мы смогли запустить только IonHammer и Pollux. Coral и Fiona работают существенно медленнее и мы не смогли дождаться завершения их работы на больших наборах чтений. Алгоритмы IonHammer и Pollux не требуют подбора параметров, необходимых для эффективной работы алгоритмов. Для Fiona требуется указывать оценку на размер генома, в качестве которой мы во всех экспериментах брали настоящую длину генома. Для оптимальной работы Coral требуется долгий подбор параметров. Для подбора параметром необходим референсный геном. В практических приложениях, для которых требуется алгоритм коррекции, референсный геном обычно отсутствует (и при наличии референсного генома нужно применять другие методы коррекции). Кроме того, долгая подборка параметров существенно увеличивает время, необходимое для получения итогового набора исправленных чтений. Поэтому для Coral мы использовали параметры по-умолчанию, за исключениям размера k-меров, который был увеличен до 31.

Для анализа качества сборки генома использовался Quast [13] версии 4.5, по-

дробные таблицы с результатами доступны в приложении Д. От хорошего алгоритма ожидается, что большая часть статистик (В первую NA50, NGA50, количество вставок/удалений/замен на 100000 нуклеотидов, процент собранного генома, количество не верно собранных участков (misassembled contigs)) будут сравнимы или лучше, чем в случае сборки без коррекции.

В результате IonHammer во всех экспериментах показал высокое качество коррекции и в большинстве случаев исправлял больше всех ошибок. Кроме того, IonHammer работал существенно быстрее всех конкурентов, а также потреблял меньше всех памяти на больших наборах данных. Качество сборки генома после коррекции с помощью IonHammer было сравнимо или превосходило качество сборки без предварительной коррекции. При этом время, необходимое на коррекцию и последующую сборку генома суммарно было меньше, чем время, необходимое на сборку генома без коррекции.

Дадим краткую характеристику качества работы алгоритмов-конкурентов.

Fiona исправляет достаточно хорошо, часто лучше, чем IonHammer. Кроме того, Fiona портит меньше всех чтений. Сборка генома на основе исправленных с помощью Fiona чтений также показывает хорошие результаты. Но использовать Fiona можно только на относительно небольших наборах данных, т.к. время работы данного алгоритма существенно больше, чем у всех остальных и на больших наборах мы не смогли дождаться завершения работы алгоритма.

Coral работает достаточно медленно, а также не способен давать хорошие результаты без предварительной подборки параметров, которую нельзя осуществить без референсного генома — при использовании параметров по-умолчанию Coral иногда исправлял хорошо, а иногда плохо. Кроме того, Coral всегда совершает систематические ошибки, существенно ухудшающие качество сборки генома (существенно увеличивает среднее число ошибок на 100000 нуклеотидов). Еще один минус Coral — ухудшение качества коррекции при увеличении числа чтений.

Pollux часто показывает сравнимые с IonHammer результаты (Pollux немного хуже), но реализация данного алгоритма однопоточная и из-за этого применять его на реальных данных также достаточно сложно — для коррекции ошибок требуется слишком много времени по сравнению с IonHammer. Кроме того, даже если предположить оптимальный масштабирование Pollux с одного потока до 32 он все равно будет медленней, чем IonHammer.

Подробные таблицы с результатами по качеству коррекций различными методами находятся в приложении Г. Сравнения скорости работы представлены в приложении В. Таблицы с статистиками по качеству сборки генома доступны в приложении Д.

Заключение

В данной работе представлена новая версия алгоритма IonHammer, предназначенного для исправления ошибок в чтениях, полученных с помощью технологии IonTorrent.

В рамках данной работы в предыдущую версию IonHammer внесены существенные изменения:

1. Существенно ускорено время работы шаг по поиску компонент связности ED_l-графа.

2. Предложен и реализован метод автоматической оценки параметров, необходимый для шага субкластеризации, а также для фильтрации ошибочных кластеров hk-меров.

3. Реализован новый метод исправления ошибок в чтениях на основе оценки множества геномных hk-меров.

Проведен анализ качества и скорости работы новой версии алгоритма. Предложенная в данной работе модификация IonHammer показывает высокое качество коррекции. Кроме того, новая версия алгоритма является наиболее быстрым и эффективным методом коррекции ошибок в данных IonTorrent.

Проведенные исследования показывают, что высокое качество работы IonHammer позволяет применять в различных задачах анализа и обработки чтений различных бактерий. Одним из таких приложений является задача сборки генома. За счет высокой скорости работы новая версия IonHammer позволяет уменьшить время, необходимое для сборки генома бактерий с помощью геномного ассемблера SPAdes, не теряя при этом в качестве.

В заключение отметим, что современные технологии не стоят на месте. Появляются новые методы секвенирования. Идеи, на которых основан IonHammer, достаточно универсальны и алгоритм коррекции можно будет адаптировать и под будущие технологии, если им будут свойственны ошибки вида «вставки и удаления».

Список литературы

1. Nikolenko S., Korobeynikov A., Alekseyev M. Bayeshammer: Bayesian clustering for error correction in single-cell sequencing // BMC Genomics. — 2013. — Vol. 14, no. 1. — P. S7. — online; accessed: http://dx.doi.org/10.1186/1471-2164-14-S1-S7.
2. Assembling single-cell genomes and mini-metagenomes from chimeric MDA products / S. Nurk, A. Bankevich, D. Antipov et al. // J. Comput. Biol. — 2013. — Oct. — Vol. 20, no. 10. — P. 714–737.
3. В. И. Левенштейн. Двоичные коды с исправлением выпадений, вставок и замещений символов // Доклады Академий Наук СССР. — 1965. — Vol. 163, no. 4. — P. 845–848.
4. Technical note. the per-base quality score system. — Access mode: http://chase.iontorrent.com/ion-docs/Technical-Note---Quality-Score_6128102.html (online; accessed: 09.05.2017).
5. Pevzner P. A., Tang H., Waterman M. S. An Eulerian path approach to DNA fragment assembly // Proceedings of the National Academy of Sciences. — 2001. — Aug. — Vol. 98, no. 17. — P. 9748–9753. — Access mode: http://dx.doi.org/10.1073/pnas.171285098.
6. Scott E., Kakaradov B. Error correction of high-throughput sequencing datasets with non-uniform coverage // Bioinformatics. — 2011. — Vol. 27, no. 13. — P. i137–41.
7. Cache-oblivious peeling of random hypergraphs / Djamal Belazzougui, Paolo Boldi, Giuseppe Ottaviano et al. // CoRR. — 2013. — Vol. abs/1312.0526. — Access mode: http://arxiv.org/abs/1312.0526.
8. Lander E. S., Waterman M. S. Genomic mapping by fingerprinting random clones: A mathematical analysis // Genomics. — 1988. — Apr. — Vol. 2, no. 3. — P. 231–239. — Access mode: http://dx.doi.org/10.1016/0888-7543(88)90007-9.
9. Tarjan R. E. Efficiency of a good but not linear set union algorithm // J. ACM. — 1975. — Apr. — Vol. 22, no. 2. — P. 215–225. — Access mode: http://doi.acm.org/10.1145/321879.321884.
10. Marinier E., Brown D. G., McConkey B. J. Pollux: platform independent error correction of single and mixed genomes // BMC Bioinformatics. — 2015. — Vol. 16, no. 1. — P. 10. — Access mode: http://dx.doi.org/10.1186/s12859-014-0435-6.
11. Salmela L., Schroder J. Correcting errors in short reads by multiple alignments //

Bioinformatics. — 2011. — Jun. — Vol. 27, no. 11. — P. 1455–1461. — Access mode: http://dx.doi.org/10.1093/bioinformatics/btr170.

12. Fiona: a parallel and automatic strategy for read error correction / M. H. Schulz, D. Weese, M. Holtgrewe et al. // Bioinformatics. — 2014. — Sep. — Vol. 30, no. 17. — P. i356–363.
13. QUAST: quality assessment tool for genome assemblies / A. Gurevich, V. Saveliev, N. Vyahhi, G. Tesler // Bioinformatics. — 2013. — Apr. — Vol. 29, no. 8. — P. 1072–1075.

Приложение А

Псевдокод алгоритмов

Algorithm 1 Вспомогательные функции.

function IsSolid(h)

h is tested hk-mer

This procedure — any estimator of set of solid hk-mers

return **True** if hk-mer estimated as good, **False** otherwise

function SkipPenalty(correction)

Input: History of previous correction

Input: non-solid hk-mer $h \in \mathcal{H}^k$

Output: penalty $p \in \mathbb{R}_{\geq 0}$

This function computes penalty for skipping non-solid hk-mer

function CorrectionPenalty(correction)

Input: History of previous correction

Input: hk-mer $h \in \mathcal{H}^k$

Input: corrected hk-mer $h_c \in \mathcal{H}^k$

Output: penalty $p \in \mathbb{R}_{\geq 0}$

This function computes penalty for correction non-solid hk-mer h to sold hk-mer h_c

function HRunCorrections(h)

Input: $h \in \mathcal{H}$

Output: Set of homopolymers — corrections of h

Parameter: m — fixed parameter, max length of possible indel. For IonTorrent data could be set to 1.

$c = \text{nucl}(h)$

$l = \text{len}(h)$

$\text{from} = \max(1, l - m)$

$\text{to} = l + m$

return {(c, from), (c, from + 1), . . . , (c, to)}

Algorithm 2 Алгоритм исправления чтений. Вспомогательные функции представлены в листинге 1

function NextStates($r \in \mathcal{H}^*$, $s \in \mathbb{R}$, $h \in \mathcal{H}$, $i \in \mathbb{N}$)

Input: r is corrected prefix of observed read

Input: s is regret suffered for corrections

Input: i is offset of h in observed read

$l \leftarrow |r| - k + 1$

$S \leftarrow \emptyset$

if IsSolid($r[l, l + k - 1)|h$) **then**

return $\{(r \mid h, i + 1, s)\}$

else

$p \leftarrow$ SkipPenalty($r[l, l + k - 1) \mid h$, $r[l, l + k - 1) \mid h$)

$S \leftarrow \{(r \mid h, i + 1, s + p)\}$

for $h_c \in$ HRunCorrections(h) **do**

if IsSolid($r[l, l + k - 1)|h_c$) **then**

$p \leftarrow$ CorrectionPenalty($r[l, l + k - 1) \mid h$, $r[l, l + k - 1) \mid h_c$)

$S \leftarrow S \cup \{(r \mid h_c, i + 1, s + p)\}$

return S

procedure Correct($r \in \mathcal{H}^*$)

Assume IsSolid($r[0, k)$) = **true**

Assume IsSolid($r[1, k + 1)$) = **false**

Let S be priority queue of (s, i, p) triples, where $s \in \mathcal{H}^*$ is a sequence and $p \in \mathbb{R}$ is its priority

$(s, i, p) \leftarrow \{(r[0, k), k, 0)\}$

$S \leftarrow \emptyset$

repeat

$S \leftarrow S \cup$ NextStates(s, p, $r[i]$, i)

$(s, i, p) \leftarrow$ Pop(S)

until $i = |r|$

return s

Приложение Б

Наборы данных

При анализе работы алгоритмов использовались наборы данных для бактерии *E. coli str. DH10B* и бактерии *E. coli str. O157H Sakai*. В статистиках hk-меров k равно 16.

Набор данных	Конфигурация оборудования	Чтений	Средняя длина	С ошибками	Геномных hk-меров	Негеномных hk-меров
B22-730	Ion 314v2, 400bp run	494921	325	390318 (78.8%)	6574862 (99.9731%)	18×10^6
R17-67	Ion 318v2, 400bp run	6837661	353	5082829 (74.3%)	6576539 (99.9986%)	13.8×10^7
DH10B-520	Ion 520, 400bp run	7247730	368	5815130 (80.0%)	6576523 (99.9983%)	20×10^7
DH10B-530	Ion 530, 400bp run	22749163	375	16809254 (73.8%)	6576603 (99.9995%)	39.7×10^7
C24-698	Ion 314v2, 200bp run	685406	244	338748 (49.4%)	6576115 (99.9921%)	10^7
SN2-158	Ion 314v2, 400bp run	649414	384	626572 (96.4%)	5991244 (91.099%)	16×10^6

Таблица Б.1. Наборы чтений для *E. coli str. DH10B*. Для геномных hk-меров в скобках указана доля от общего числа геномных hk-меров.

Набор данных	Конфигурация оборудования	Чтений	Средняя длина	С ошибками	Геномных hk-меров	Негеномных hk-меров
C23-836	Ion PGM Hi-Q, 425bp, Ion 314	510359	364	320983 (62.8%)	7669625 (99.9623%)	15.6×10^6
C24-835	Ion PGM Hi-Q, 425bp, Ion 318	6500837	361	4384940 (67.4%)	7670316 (99.9713%)	14×10^7
BEA-1108	Ion PGM Hi-Q, 350bp, Ion 314	804397	330	322707 (40.1%)	7669821 (99.9648%)	12.7×10^6
BEA-1107	Ion PGM Hi-Q, 350bp, Ion 314	7438086	322	4016155 (53.9%)	7670267 (99.9707%)	12×10^7

Таблица Б.2. Наборы чтений для *E. coli str. O157H Sakai*.

Приложение B

Сравнение времени работы алгоритмов

При сравнении скорости Coral запускался в 8 потоков, Pollux в 1 поток, IonHammer и Fiona в 32 потока. Результаты измерения скорости работы представлены в таблице B.1 и таблице B.2. Во всех случаях IonHammer показа наиболее высокую скорость работы. При этом расход памяти у IonHammer также был достаточно низкий — на наборе данных *E. coli DH10B*-520 IonHammer использовал меньше всех памяти. При этом, большой расход памяти на наборе с небольшим количеством чтений *E. coli 0157H Sakai*-1108 вызван тем, что для оптимизации скорости работы алгоритма на каждый поток всегда выделятся достаточно большое число оперативной памяти под временные данные.

После коррекции мы также измерели время сборки генома с помощью SPAdes версии 3.10.1. Результаты данных замеров представлены в таблице B.3 и таблице B.2.

	IonHammer (32)	Pollux (1)	Coral (8)	Fiona (32)
Процессорное время	**29298s**	65866s	439084s	∞
Время работы	**0:17:33**	18:19:48	15:46:34	∞
Максимальный RSS (GB)	**15**	20	33	NA

Таблица B.1. Време работы на наборе данных *E. coli DH10B*-520.

	IonHammer (32)	Pollux (1)	Coral (8)	Fiona (32)
Процессорное время	**2835s**	4123s	31521s	86276
Время работы	**0:2:15**	1:08:57	1:12:09	0:51:38
Максимальный RSS (GB)	15	**2**	5	**2**

Таблица B.2. Време работы на наборе данных *E. coli 0157H Sakai*-1108.

	IonHammer	Pollux	Coral	Fiona	Without
Процессорное время	**8886s**	10104s	18005s	NA	26043s
Время работы	**0:8:06**	0:12:11	0:44:55	NA	1:27:16
RSS	**16**	16	22	NA	29

Таблица В.3. Време сборки генома для набора данных *E. coli DH10B*-520.

	IonHammer	Pollux	Coral	Fiona	Without
Процессорное время	1189s	1150s	1232s	**1120s**	1763s
Время работы	0:1:40	0:1:34	0:1:56	**0:1:29**	0:4:22
RSS	4	4	4	3	3

Таблица В.4. Време сборки генома для набора данных *E. coli 0157H Sakai*-1108.

Приложение Г

Таблицы с качеством коррекции чтений

Для измерения качества чтения и исправленные чтения, полученные с помощью различных алгоритмов, выравнивались на референсный геном с помощью программы для выравнивания чтений TMAP, входящей в состав программного комплекса, поставляемого вместе с оборудованием IonTorrent. В каждой таблице в скобках указана доля чтений от общего числа чтений. Для расстояние Левенштейна в скобках указано относительно улучшение расстояния (если до коррекции среднее расстояние равнялось x, а после y, то в скобках будет указано $\frac{x}{y}$). Для количества геномных hk-меров в скобках указана доля от общего числа hk-меров в геноме. Для количества негеномных hk-меров — отношение $\frac{x}{y}$, где x равно количеству негеномных hk-меров до коррекции, y количеству негеномных hk-меров после коррекции.

Таблица Г.1. Сравнение старой и новой версии IonHammer, *E. coli str. DH10B*-C24.

Алгоритм коррекции	New IonHammer	Old IonHammer
Сравнимые чтения	658215 (96.26%)	655275 (95.83%)
Испорченные чтения	8085 (1.18%)	12582 (1.84%)
Полностью исправленные чтения	266919 (39.04%)	154200 (22.55%)
Исправленные чтения	299417 (43.79%)	239966 (35.09%)
Среднее расстояние	0.2542 (x4.86)	0.6755 (x1.83)
Испорченные, с одной ошибкой	826 (0.12%)	3621 (0.53%)
Исправленные, с 1 ошибкой	131762 (19.27%)	102019 (14.92%)
Испорченные и изменения выравнивания	33676 (4.92%)	40037 (5.86%)
Изменения выравнивания	25591 (3.74%)	27455 (4.02%)
Геномные hk-меры	6553831 (99.65%)	6567815 (99.86%)
Негеномные hk-меры	716331	3661340

Таблица Г.2. Сравнение старой и новой версии IonHammer, *E. coli str. DH10B*-SN2-158.

Алгоритм коррекции	New IonHammer	Old IonHammer
Сравнимые чтения	624977 (96.31%)	623177 (96.03%)
Испорченные чтения	5906 (0.91%)	31153 (4.80%)
Полностью исправленные чтения	95307 (14.69%)	57510 (8.86%)
Исправленные чтения	548802 (84.57%)	450447 (69.41%)
Среднее расстояние	2.1570 (x2.28)	3.5669 (x1.38)
Испорченные, с 1 ошибкой	237 (0.04%)	662 (0.10%)
Исправленные, с 1 ошибкой	23696 (3.65%)	22390 (3.45%)
Испорченные и изменения выравнивания	29977 (4.62%)	54757 (8.44%)
Изменения выравнивания	24071 (3.71%)	23604 (3.64%)
Геномные hk-меры	5990612 (91.0893%)	5990664 (91.0901%)
Негеномные hk-меры	2402470 (x6.6)	6355675 (x2.5)

Таблица Г.3. *E. coli str. DH10B*-314

Алгоритм коррекции	Fiona	Pollux	Coral	IonHammer
Сравнимые чтения	472844 (96.42%)	472680 (96.38%)	472748 (96.40%)	473210 (96.49%)
Испорченные чтения	1686 (0.34%)	3586 (0.73%)	3531 (0.72%)	3091 (0.63%)
Полностью исправленные чтения	272220 (55.51%)	256651 (52.33%)	310841 (63.38%)	298797 (60.93%)
Исправленные чтения	355714 (72.53%)	331730 (67.64%)	345390 (70.43%)	355611 (72.51%)
Среднее расстояние	0.8607 (x3.92)	1.2713 (x2.65)	0.8682 (x3.88)	0.9244 (x3.65)
Испорченные, с 1 ошибкой	309 (0.06%)	578 (0.12%)	472 (0.10%)	508 (0.10%)
Исправленные, с 1 ошибкой	80825 (16.48%)	82801 (16.88%)	84634 (17.26%)	87121 (17.76%)
Испорченные и изменения выравнивания	19597 (4.00%)	21408 (4.37%)	21200 (4.32%)	20396 (4.16%)
Изменения выравнивания	17911 (3.65%)	17822 (3.63%)	17669 (3.60%)	17305 (3.53%)
Геномные hk-меры	6567738 (99.86%)	6552954 (99.64%)	6561386 (99.76%)	6559970 (99.74%)
Негеномные hk-меры	1379635 (x13)	1750615 (x10)	2875518 (x6)	1268579 (x14)

Таблица Г.4. *E. coli str. DH10B*-318

Алгоритм коррекции	Pollux	Coral	IonHammer
Сравнимые чтения	6569922 (96.34%)	6576227 (96.43%)	6578927 (96.47%)
Испорченные чтения	31180 (0.46%)	271235 (3.98%)	53625 (0.79%)
Полностью исправленные чтения	3951980 (57.95%)	1608163 (23.58%)	4140555 (60.71%)
Исправленные чтения	4556540 (66.81%)	2757208 (40.43%)	4576141 (67.10%)
Среднее расстояние	0.8268 (x3.36)	1.6308 (x1.70)	0.7452 (x3.73)
Испорченные, с 1 ошибкой	4935 (0.07%)	80474 (1.18%)	10906 (0.16%)
Исправленные, с 1 ошибкой	1379570 (20.23%)	536692 (7.87%)	1399203 (20.52%)
Испорченные и изменения выравнивания	281676 (4.13%)	514906 (7.55%)	295496 (4.33%)
Изменения выравнивания	250496 (3.67%)	243671 (3.57%)	241871 (3.55%)
Геномные hk-меры	6575275 (99.97%)	6574309 (99.96%)	6576451 (99.99%)
Негеномные hk-меры	18307450 (x7.5)	63574575 (x2.2)	21066804 (x6.5)

Таблица Г.5. *E. coli str. DH10B*-520

Алгоритм коррекции	Pollux	Coral	IonHammer
Сравнимые чтения	6982854 (96.44%)	6988067 (96.51%)	6991291 (96.56%)
Испорченные чтения	23072 (0.32%)	89576 (1.24%)	40037 (0.55%)
Полностью исправленные чтения	4401868 (60.79%)	726071 (10.03%)	4829478 (66.70%)
Исправленные чтения	5319467 (73.47%)	2514864 (34.73%)	5389892 (74.44%)
Среднее расстояние	1.1725 (x3.37)	2.5868 (x1.53)	0.9411 (x4.20)
Испорченные, с 1 ошибкой	3205 (0.04%)	28178 (0.39%)	6656 (0.09%)
Исправленные, с 1 ошибкой	1211238 (16.73%)	185152 (2.56%)	1229701 (16.98%)
Испорченные и изменения выравнивания	282019 (3.89%)	342169 (4.73%)	291321 (4.02%)
Изменения выравнивания	258947 (3.58%)	252593 (3.49%)	251284 (3.47%)
Геномные hk-меры	6576079 (99.9916%)	6576221 (99.9937%)	6576353 (99.9957%)
Негеномные hk-меры	31411859 (x6.3)	114739308 (x1.75)	24623561 (x8.1)

Таблица Г.6. *E. coli str. DH10B*-530

Алгоритм коррекции	Pollux	IonHammer
Сравнимые чтения	21915391 (96.44%)	21940093 (96.54%)
Испорченные чтения	135852 (0.60%)	229238 (1.01%)
Полностью исправленные чтения	12849560 (56.54%)	13408091 (59.00%)
Исправленные чтения	15223056 (66.99%)	15325018 (67.44%)
Среднее расстояние	1.0472 (x3.29)	0.9055 (x3.80)
Испорченные, с 1 ошибкой	13957 (0.06%)	37429 (0.16%)
Исправленные, с 1 ошибкой	3947107 (17.37%)	3930078 (17.29%)
Испорченные и изменения выравнивания	948696 (4.17%)	1019608 (4.49%)
Изменения выравнивания	812844 (3.58%)	790370 (3.48%)
Геномные hk-меры	6576213 (99.9936%)	6576491 (99.9978%)
Негеномные hk-меры	66899839 (x5.9)	57876110 (x6.8)

Таблица Г.7. *E. coli str. DH10B*-C24

Алгоритм коррекции	Fiona	Pollux	Coral	IonHammer
Сравнимые чтения	658354 (96.28%)	657360 (96.14%)	656476 (96.01%)	658553 (96.31%)
Испорченные чтения	1003 (0.15%)	8150 (1.19%)	9698 (1.42%)	7875 (1.15%)
Полностью исправленные чтения	237367 (34.71%)	235284 (34.41%)	281945 (41.23%)	266909 (39.03%)
Исправленные чтения	280627 (41.04%)	277966 (40.65%)	300629 (43.97%)	299172 (43.75%)
Среднее расстояние	0.2517 (x4.91)	0.4032 (x3.07)	0.2299 (x5.38)	0.2579 (x4.79)
Испорченные, с 1 ошибкой	200 (0.03%)	982 (0.14%)	779 (0.11%)	771 (0.11%)
Исправленные, с 1 ошибкой	111199 (16.26%)	125778 (18.39%)	133384 (19.51%)	131801 (19.28%)
Испорченные и изменения выравнивания	26530 (3.88%)	34589 (5.06%)	37002 (5.41%)	33126 (4.84%)
Изменения выравнивания	25527 (3.73%)	26439 (3.87%)	27304 (3.99%)	25251 (3.69%)
Геномные hk-меры	6572520 (99.9375%)	6557985 (99.7165%)	6548650 (99.5745%)	6553771 (99.6524%)
Негеномные hk-меры	745943 (x13.4)	1578382 (x6.3)	827297 (x12)	726543 (x13.7)

Таблица Г.8. *E. coli str. O157H Sakai*-BEA1107

Алгоритм коррекции	Pollux	Coral	IonHammer
Сравнимые чтения	7057897 (97.50%)	7074481 (97.73%)	7075206 (97.74%)
Испорченные чтения	48334 (0.67%)	169774 (2.35%)	54250 (0.75%)
Полностью исправленные чтения	3157486 (43.62%)	581222 (8.03%)	3234301 (44.68%)
Исправленные чтения	3584550 (49.52%)	1025809 (14.17%)	3533981 (48.82%)
Среднее расстояние	0.9546 (x2.27)	1.8453 (x1.17)	0.9252 (x2.34)
Испорченные, с 1 ошибкой	10641 (0.15%)	36811 (0.51%)	17118 (0.24%)
Исправленные, с 1 ошибкой	1409115 (19.47%)	250336 (3.46%)	1412389 (19.51%)
Испорченные и изменения выравнивания	230066 (3.18%)	334172 (4.62%)	218931 (3.02%)
Изменения выравнивания	181732 (2.51%)	164398 (2.27%)	164681 (2.27%)
Геномные hk-меры	7669601 (99.962%)	7668747 (99.9509%)	7669843 (99.9651%)
Негеномные hk-меры	18527404 (x6.5)	90476633 (x1.3)	23625124 (x5)

Таблица Г.9. *E. coli str. O157H Sakai*-BEA1108

Алгоритм коррекции	Fiona	Pollux	Coral	IonHammer
Сравнимые чтения	765442 (97.72%)	764201 (97.56%)	763795 (97.51%)	766123 (97.81%)
Испорченные чтения	1376 (0.18%)	6008 (0.77%)	138225 (17.65%)	7054 (0.90%)
Полностью исправленные чтения	262798 (33.55%)	243265 (31.06%)	202357 (25.83%)	244015 (31.15%)
Исправленные чтения	294233 (37.56%)	278025 (35.49%)	240837 (30.75%)	269118 (34.36%)
Среднее расстояние	0.6858 (x2.56)	0.9440 (x1.86)	1.2626 (x1.39)	0.9196 (x1.91)
Испорченные, с 1 ошибкой	151 (0.02%)	946 (0.12%)	13412 (1.71%)	2334 (0.30%)
Исправленные, с 1 ошибкой	119999 (15.32%)	122002 (15.58%)	93223 (11.90%)	119335 (15.24%)
Испорченные и изменения выравнивания	19483 (2.49%)	25177 (3.21%)	157728 (20.14%)	24296 (3.10%)
Изменения выравнивания	18107 (2.31%)	19169 (2.45%)	19503 (2.49%)	17242 (2.20%)
Геномные hk-меры	7668449 (99.947%)	7666769 (99.9251%)	7600326 (99.0591%)	7668689 (99.9501%)
Негеномные hk-меры	1303469 (x9.7)	2294771 (x5.5)	2754396 (x4.6)	2756551 (x4.6)

Таблица Г.10. *E. coli str. O157H Sakai*-C23

Алгоритм коррекции	Fiona	Pollux	Coral	IonHammer
Сравнимые чтения	482865 (97.61%)	481544 (97.35%)	239715 (48.46%)	482978 (97.64%)
Испорченные чтения	579 (0.12%)	3675 (0.74%)	29550 (5.97%)	3876 (0.78%)
Полностью исправленные чтения	270550 (54.69%)	247383 (50.01%)	104007 (21.03%)	265537 (53.68%)
Исправленные чтения	301100 (60.87%)	286459 (57.91%)	131963 (26.68%)	291025 (58.83%)
Среднее расстояние	0.3133 (x6.71)	0.5610 (x3.74)	0.7235 (x2.90)	0.4481 (x4.69)
Испорченные, с 1 ошибкой	132 (0.03%)	849 (0.17%)	3726 (0.75%)	1022 (0.21%)
Исправленные, с 1 ошибкой	97235 (19.66%)	97547 (19.72%)	36799 (7.44%)	99517 (20.12%)
Испорченные и изменения выравнивания	12512 (2.53%)	16818 (3.40%)	37158 (7.51%)	15598 (3.15%)
Изменения выравнивания	11933 (2.41%)	13143 (2.66%)	7608 (1.54%)	11722 (2.37%)
Геномные hk-меры	7667430 (99.9337%)	7664870 (99.9003%)	7555354 (98.4729%)	7666935 (99.9272%)
Негеномные hk-меры	882714 (x17.6)	1917820 (x8.1)	951508 (x16.4)	1801649 (x8.7)

Таблица Г.11. *E. coli str. O157H Sakai*-C24

Алгоритм коррекции	Pollux	Coral	IonHammer
Сравнимые чтения	6138702 (97.27%)	6150211 (97.45%)	6154966 (97.53%)
Испорченные чтения	39707 (0.63%)	865140 (13.71%)	39004 (0.62%)
Полностью исправленные чтения	3502557 (55.50%)	2616323 (41.46%)	3713324 (58.84%)
Исправленные чтения	3976575 (63.01%)	3267936 (51.78%)	4020455 (63.70%)
Среднее расстояние	0.5580 (x4.33)	1.0780 (x2.24)	0.4543 (x5.32)
Испорченные, с 1 ошибкой	9766 (0.15%)	202343 (3.21%)	9836 (0.16%)
Исправленные, с 1 ошибкой	1241745 (19.68%)	840044 (13.31%)	1266029 (20.06%)
Испорченные и изменения выравнивания	212419 (3.37%)	1026139 (16.26%)	195526 (3.10%)
Изменения выравнивания	172712 (2.74%)	160999 (2.55%)	156522 (2.48%)
Геномные hk-меры	7669284 (99.9578%)	7641012 (99.5894%)	7669858 (99.9653%)
Негеномные hk-меры	19465920 (x7.2)	24431022 (x5.7)	19404183 (x7.2)

Приложение Д

Результаты сборки генома

Для сборки генома использовался SPAdes версии 3.10.1. Таблицы с качеством сборки генома построены с помощью Quast версии 4.5. Все статистики посчитаны для контигов (contigs) сразмера $\geq$ 500 bp, если не указано иного (например, "# contigs ($\geq$ 0 bp)"или "Total length ($\geq$ 0 bp)" включают все контиги).

Таблица Д.1. *E. coli str. DH10B*-314

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# contigs (≥ 0 bp)	143	142	161	324	146
# contigs (≥ 1000 bp)	101	97	99	94	101
# contigs (≥ 5000 bp)	77	71	73	69	73
# contigs (≥ 10000 bp)	70	65	67	63	66
# contigs (≥ 25000 bp)	61	58	59	56	55
# contigs (≥ 50000 bp)	34	33	33	33	32
Total length (≥ 0 bp)	4473946	4473615	4477291	**4493615**	4475753
Total length (≥ 1000 bp)	4459587	4458474	**4464513**	4456897	4462831
Total length (≥ 5000 bp)	4401119	4392638	**4403154**	4397049	4396640
Total length (≥ 10000 bp)	4345422	4346637	**4357073**	4350976	4341349
Total length (≥ 25000 bp)	4202016	4231284	4224561	**4235366**	4161424
Total length (≥ 50000 bp)	3250502	3343935	3282146	**3403103**	3332138
# contigs	111	108	107	105	109
Largest contig	269966	270180	**326947**	269608	326884
Total length	4467075	4466649	**4470624**	4465062	4468819
Reference length	4686137	4686137	4686137	4686137	4686137
GC (%)	50.73	50.73	50.73	50.73	50.73
Reference GC (%)	50.78	50.78	50.78	50.78	50.78
N50	74484	82853	74473	**87017**	85557
NG50	71491	74434	72817	**85559**	82854
N75	45583	48394	47492	**53131**	48397
NG75	42278	43045	43036	**45583**	43506
L50	18	**15**	17	**15**	**15**
LG50	19	17	18	**16**	**16**
L75	37	34	35	**32**	33
LG75	40	37	39	**36**	**36**
# misassemblies	1	**0**	1	**0**	**0**
# misassembled contigs	1	**0**	1	**0**	**0**
Misassembled contigs length	119618	**0**	72817	**0**	**0**
# local misassemblies	**20**	25	**20**	26	27
# unaligned mis. contigs	0	0	0	0	0
# unaligned contigs	0 + 0 part	0 + 0 part	0 + 0 part	0 + 0 part	0 + 0 part
Unaligned length	0	0	0	0	0
Genome fraction (%)	95.298	95.262	**95.351**	95.267	95.337
Duplication ratio	**1**	1.001	1.001	**1**	**1**
# N's per 100 kbp	0.00	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	6.65	5.49	6.58	**5.00**	5.93
# indels per 100 kbp	19.68	19.67	26.92	18.70	**18.69**
Largest alignment	269966	270180	**326947**	269594	326884
Total aligned length	4466781	4465002	**4468895**	4464834	4468590
NA50	74484	76394	74473	**87017**	85557
NGA50	71491	74434	71483	**85559**	82854
NA75	45583	48394	47492	**53131**	48397
NGA75	42278	43045	43036	**45583**	43506
LA50	18	16	17	**15**	**15**
LGA50	19	17	18	**16**	**16**
LA75	37	34	35	**32**	33
LGA75	40	37	39	**36**	**36**

Таблица Д.2. *E. coli str. DH10B*-314

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# misassemblies	1	**0**	1	**0**	**0**
# relocations	1	**0**	1	**0**	**0**
# translocations	0	0	0	0	0
# inversions	0	0	0	0	0
# misassembled contigs	1	**0**	1	**0**	**0**
Misassembled contigs length	119618	**0**	72817	**0**	**0**
# local misassemblies	**20**	25	**20**	26	27
# unaligned mis. contigs	0	0	0	0	0
# mismatches	297	245	294	**223**	265
# indels	879	878	1203	**835**	**835**
# indels ($\leq$ 5 bp)	879	877	1203	835	**833**
# indels ($>$ 5 bp)	**0**	1	**0**	**0**	2
Indels length	907	927	1233	**856**	947

Таблица Д.3. *E. coli str. DH10B*-318

Assembly	Raw reads	Pollux	Coral	IonHammer
# contigs (≥ 0 bp)	157	132	146	150
# contigs (≥ 1000 bp)	88	88	84	87
# contigs (≥ 5000 bp)	65	64	64	63
# contigs (≥ 10000 bp)	60	57	58	56
# contigs (≥ 25000 bp)	53	51	52	51
# contigs (≥ 50000 bp)	33	30	32	30
Total length (≥ 0 bp)	4480725	4477490	**4483501**	4480525
Total length (≥ 1000 bp)	4463343	**4465640**	4464018	4464695
Total length (≥ 5000 bp)	4415303	4415361	**4422902**	4418468
Total length (≥ 10000 bp)	4373785	4360551	**4376269**	4363636
Total length (≥ 25000 bp)	4268808	4270820	**4287920**	4287756
Total length (≥ 50000 bp)	3553632	3518229	**3572143**	3530861
# contigs	97	95	97	96
Largest contig	269972	**327186**	270096	326561
Total length	4469978	4470522	**4473486**	4470652
Reference length	4686137	4686137	4686137	4686137
GC (%)	50.73	50.73	50.73	50.73
Reference GC (%)	50.78	50.78	50.78	50.78
N50	92103	102216	96576	**107008**
NG50	88613	96716	92074	**96719**
N75	56523	**57774**	56614	57773
NG75	50565	53136	**53161**	53137
L50	15	14	14	**13**
LG50	16	15	15	**14**
L75	30	28	28	**27**
LG75	33	**30**	31	**30**
# misassemblies	1	1	1	**0**
# misassembled contigs	1	1	1	**0**
Misassembled contigs length	50565	102216	50230	**0**
# local misassemblies	22	23	**21**	25
# unaligned mis. contigs	0	0	0	0
# unaligned contigs	1 + 0 part	**0 + 0 part**	1 + 0 part	1 + 0 part
Unaligned length	765	**0**	765	765
Genome fraction (%)	95.314	**95.377**	95.353	95.363
Duplication ratio	1.001	**1**	1.001	**1**
# N's per 100 kbp	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	**4.39**	4.94	5.55	4.92
# indels per 100 kbp	6.69	6.35	39.21	**6.11**
Largest alignment	269861	**327186**	269985	326561
Total aligned length	4467464	4470375	**4470959**	4469787
NA50	92103	96716	96576	**107008**
NGA50	88613	92054	92074	**96719**
NA75	56523	**57774**	56588	57773
NGA75	47465	47465	**53161**	53137
LA50	15	14	14	**13**
LGA50	16	15	15	**14**
LA75	30	28	28	**27**
LGA75	33	31	31	**30**

Таблица Д.4. *E. coli str. DH10B*-318

Assembly	Raw reads	Pollux	Coral	IonHammer
# misassemblies	1	1	1	**0**
# relocations	1	1	1	**0**
# translocations	0	0	0	0
# inversions	0	0	0	0
# misassembled contigs	1	1	1	**0**
Misassembled contigs length	50565	102216	50230	**0**
# local misassemblies	22	23	**21**	25
# unaligned mis. contigs	0	0	0	0
# mismatches	**196**	221	248	220
# indels	299	284	1752	**273**
# indels ($\leq$ 5 bp)	299	284	1751	**273**
# indels ($>$ 5 bp)	**0**	**0**	1	**0**
Indels length	307	305	1927	**283**

Таблица Д.5. *E. coli str. DH10B*-520

Assembly	Raw reads	Pollux	Coral	IonHammer
# contigs (≥ 0 bp)	144	133	148	159
# contigs (≥ 1000 bp)	98	97	99	98
# contigs (≥ 5000 bp)	67	66	67	66
# contigs (≥ 10000 bp)	59	56	59	57
# contigs (≥ 25000 bp)	51	51	52	52
# contigs (≥ 50000 bp)	31	30	32	30
Total length (≥ 0 bp)	4506676	**4509982**	4508005	4508136
Total length (≥ 1000 bp)	4492073	**4496841**	4492688	4493015
Total length (≥ 5000 bp)	4420866	**4429565**	4419959	4425279
Total length (≥ 10000 bp)	**4360745**	4355918	4359844	4359071
Total length (≥ 25000 bp)	4242371	4279815	4254275	**4282795**
Total length (≥ 50000 bp)	3527202	3531202	**3538976**	3498146
# contigs	110	109	111	106
Largest contig	326308	**327188**	269596	326729
Total length	4500462	**4505766**	4501035	4499102
Reference length	4686137	4686137	4686137	4686137
GC (%)	50.72	50.72	50.72	50.72
Reference GC (%)	50.78	50.78	50.78	50.78
N50	92129	**107015**	92139	96856
NG50	88836	**96857**	88623	92129
N75	56525	56524	**56530**	56526
NG75	53138	53138	**53139**	47466
L50	14	**13**	15	14
LG50	15	**14**	16	15
L75	29	**28**	30	**28**
LG75	31	**30**	32	31
# misassemblies	1	**0**	1	**0**
# misassembled contigs	1	**0**	1	**0**
Misassembled contigs length	247031	**0**	247062	**0**
# local misassemblies	19	23	**18**	22
# unaligned mis. contigs	0	0	0	0
# unaligned contigs	**10 + 1 part**	11 + 2 part	**10 + 1 part**	12 + 1 part
Unaligned length	26793	29165	**26787**	28261
Genome fraction (%)	95.400	**95.496**	95.409	95.348
Duplication ratio	1.001	**1**	1.001	1.001
# N's per 100 kbp	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	4.61	4.83	**4.18**	5.84
# indels per 100 kbp	**2.73**	2.90	11.88	2.84
Largest alignment	326308	**327188**	269566	326729
Total aligned length	4472032	**4476539**	4472590	4469219
NA50	92129	**107015**	92139	96856
NGA50	87017	**96857**	88623	92129
NA75	56525	56524	**56530**	56526
NGA75	47466	**53138**	47472	47466
LA50	14	**13**	15	14
LGA50	16	**14**	16	15
LA75	29	**28**	30	**28**
LGA75	32	**30**	33	31

Таблица Д.6. *E. coli str. DH10B*-520

Assembly	Raw reads	Pollux	Coral	IonHammer
# misassemblies	1	**0**	1	**0**
# relocations	1	**0**	1	**0**
# translocations	0	0	0	0
# inversions	0	0	0	0
# misassembled contigs	1	**0**	1	**0**
Misassembled contigs length	247031	**0**	247062	**0**
# local misassemblies	19	23	**18**	22
# unaligned mis. contigs	0	0	0	0
# mismatches	206	216	**187**	261
# indels	**122**	130	531	127
# indels ($\leq$ 5 bp)	**122**	129	531	127
# indels ($>$ 5 bp)	**0**	1	**0**	**0**
Indels length	**131**	140	571	133

Таблица Д.7. *E. coli str. DH10B*-530

Assembly	Raw reads	Pollux	IonHammer
# contigs (≥ 0 bp)	2691	275	362
# contigs (≥ 1000 bp)	208	98	100
# contigs (≥ 5000 bp)	141	68	70
# contigs (≥ 10000 bp)	109	59	60
# contigs (≥ 25000 bp)	71	52	50
# contigs (≥ 50000 bp)	27	31	29
Total length (≥ 0 bp)	**4785804**	4538895	4547419
Total length (≥ 1000 bp)	4474106	**4502859**	4497628
Total length (≥ 5000 bp)	4308675	**4433805**	4430050
Total length (≥ 10000 bp)	4061839	**4364657**	4356138
Total length (≥ 25000 bp)	3458276	**4258119**	4204427
Total length (≥ 50000 bp)	1883420	**3501688**	3421353
# contigs	257	118	125
Largest contig	110565	284532	**327256**
Total length	4511383	**4518881**	4517500
Reference length	4686137	4686137	4686137
GC (%)	50.74	50.73	50.73
Reference GC (%)	50.78	50.78	50.78
N50	41975	96879	**96966**
NG50	41562	**92910**	92644
N75	26951	**56701**	53632
NG75	24206	**47462**	**47462**
L50	36	**14**	**14**
LG50	38	**15**	**15**
L75	69	**29**	**29**
LG75	74	32	**31**
# misassemblies	**0**	1	**0**
# misassembled contigs	**0**	1	**0**
Misassembled contigs length	**0**	284532	**0**
# local misassemblies	**14**	18	20
# unaligned mis. contigs	0	0	0
# unaligned contigs	24 + 0 part	**23 + 0 part**	23 + 1 part
Unaligned length	**41604**	43228	43412
Genome fraction (%)	95.080	**95.467**	**95.467**
Duplication ratio	1.003	**1**	**1**
# N's per 100 kbp	0.00	0.00	0.00
# mismatches per 100 kbp	**1.66**	3.20	2.84
# indels per 100 kbp	15.31	**12.61**	15.83
Largest alignment	110565	269742	**327256**
Total aligned length	4469773	**4473985**	4473984
NA50	41975	96879	**96966**
NGA50	41562	**92910**	92644
NA75	26951	**54859**	53632
NGA75	24206	**47462**	**47462**
LA50	36	**14**	**14**
LGA50	38	**15**	**15**
LA75	69	30	**29**
LGA75	74	32	**31**

Таблица Д.8. *E. coli str. DH10B*-530

Assembly	Raw reads	Pollux	IonHammer
# misassemblies	**0**	1	**0**
# relocations	**0**	1	**0**
# translocations	0	0	0
# inversions	0	0	0
# misassembled contigs	**0**	1	**0**
Misassembled contigs length	**0**	284532	**0**
# local misassemblies	**14**	18	20
# unaligned mis. contigs	0	0	0
# mismatches	**74**	143	127
# indels	682	**564**	708
# indels ($\leq$ 5 bp)	682	**563**	708
# indels ($>$ 5 bp)	**0**	1	**0**
Indels length	685	**573**	715

Таблица Д.9. *E. coli str. DH10B*-C24

Assembly	Raw reads	Pollux	Coral	IonHammer
# contigs (≥ 0 bp)	203	189	172	210
# contigs (≥ 1000 bp)	100	102	100	98
# contigs (≥ 5000 bp)	73	73	73	73
# contigs (≥ 10000 bp)	66	66	66	66
# contigs (≥ 25000 bp)	55	55	57	57
# contigs (≥ 50000 bp)	31	31	32	32
Total length (≥ 0 bp)	4484030	4480675	4481115	**4486023**
Total length (≥ 1000 bp)	4458767	4457029	**4463368**	4458295
Total length (≥ 5000 bp)	4398704	4396512	**4403576**	4400496
Total length (≥ 10000 bp)	4343821	4341600	**4348507**	4345359
Total length (≥ 25000 bp)	4192560	4187110	**4225062**	4221686
Total length (≥ 50000 bp)	3358545	3352012	**3358793**	3358172
# contigs	109	113	107	108
Largest contig	326760	326847	326129	**327122**
Total length	4465655	4465540	**4469239**	4465947
Reference length	4686137	4686137	4686137	4686137
GC (%)	50.73	50.73	50.73	50.73
Reference GC (%)	50.78	50.78	50.78	50.78
N50	**88603**	**88603**	87011	87000
NG50	**87014**	86997	85547	85539
N75	**53131**	53126	53126	53129
NG75	43252	43039	43039	**43459**
L50	**15**	**15**	16	16
LG50	**16**	**16**	17	17
L75	**31**	**31**	32	32
LG75	**35**	**35**	36	36
# misassemblies	**0**	1	**0**	**0**
# misassembled contigs	**0**	1	**0**	**0**
Misassembled contigs length	**0**	29945	**0**	**0**
# local misassemblies	18	18	**13**	15
# unaligned mis. contigs	0	0	0	0
# unaligned contigs	0 + 0 part	0 + 0 part	0 + 0 part	0 + 0 part
Unaligned length	0	0	0	0
Genome fraction (%)	95.285	95.280	**95.334**	95.296
Duplication ratio	1	1	1	1
# N's per 100 kbp	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	**2.84**	3.54	2.98	3.25
# indels per 100 kbp	**23.49**	24.48	26.46	23.87
Largest alignment	326760	326847	326129	**327122**
Total aligned length	4465593	4465409	**4467717**	4465943
NA50	**88603**	**88603**	87011	87000
NGA50	**87014**	86997	85547	85539
NA75	**53131**	53126	53126	53129
NGA75	43252	43039	43039	**43459**
LA50	**15**	**15**	16	16
LGA50	**16**	**16**	17	17
LA75	**31**	**31**	32	32
LGA75	**35**	**35**	36	36

Таблица Д.10. *E. coli str. DH10B*-C24

Assembly	Raw reads	Pollux	Coral	IonHammer
# misassemblies	**0**	1	**0**	**0**
# relocations	**0**	1	**0**	**0**
# translocations	0	0	0	0
# inversions	0	0	0	0
# misassembled contigs	**0**	1	**0**	**0**
Misassembled contigs length	**0**	29945	**0**	**0**
# local misassemblies	18	18	**13**	15
# unaligned mis. contigs	0	0	0	0
# mismatches	**127**	158	133	145
# indels	**1049**	1093	1182	1066
# indels ($\leq$ 5 bp)	**1049**	1092	1182	1066
# indels ($>$ 5 bp)	**0**	1	**0**	**0**
Indels length	**1082**	1173	1210	1106

Таблица Д.11. *E. coli str. O157H Sakai*-BEA1107

Assembly	Raw reads	Pollux	Coral	IonHammer
# contigs (≥ 0 bp)	467	457	442	487
# contigs (≥ 1000 bp)	151	147	149	155
# contigs (≥ 5000 bp)	75	73	71	74
# contigs (≥ 10000 bp)	55	56	56	57
# contigs (≥ 25000 bp)	45	47	45	46
# contigs (≥ 50000 bp)	29	26	28	27
Total length (≥ 0 bp)	5361343	5359755	5360475	**5369995**
Total length (≥ 1000 bp)	5256694	5251698	**5263518**	5254930
Total length (≥ 5000 bp)	5090924	**5094141**	5091781	5079508
Total length (≥ 10000 bp)	4953225	4978077	**4991418**	4959727
Total length (≥ 25000 bp)	4764220	**4819947**	4797685	4768828
Total length (≥ 50000 bp)	4161096	4045646	**4196949**	4075995
# contigs	213	219	212	235
Largest contig	374899	374900	**374980**	374899
Total length	5298908	5301337	5307196	**5309535**
Reference length	5498450	5498450	5498450	5498450
GC (%)	50.27	50.28	50.27	50.28
Reference GC (%)	50.54	50.54	50.54	50.54
N50	144022	148746	148714	**148898**
NG50	144022	146450	**146467**	146449
N75	66756	66757	**73590**	66574
NG75	56452	47590	**66778**	49924
L50	13	**12**	**12**	**12**
LG50	13	13	13	13
L75	26	**25**	**25**	26
LG75	29	28	**27**	28
# misassemblies	2	1	**0**	**0**
# misassembled contigs	2	1	**0**	**0**
Misassembled contigs length	252939	6225	**0**	**0**
# local misassemblies	21	17	20	**13**
# unaligned mis. contigs	0	0	0	0
# unaligned contigs	8 + 2 part	**7 + 2 part**	7 + 3 part	9 + 3 part
Unaligned length	97033	98211	**96990**	97860
Genome fraction (%)	94.534	94.558	94.670	**94.709**
Duplication ratio	1.001	1.001	1.001	1.001
# N's per 100 kbp	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	**10.58**	14.50	11.24	13.27
# indels per 100 kbp	**1.52**	1.92	16.87	1.71
Largest alignment	374899	374900	**374980**	374899
Total aligned length	5200711	5201979	5209052	**5210568**
NA50	137748	148746	148714	**148898**
NGA50	137748	145451	**145468**	145450
NA75	66171	66757	**73590**	66574
NGA75	56452	47590	**66778**	49924
LA50	13	**12**	**12**	**12**
LGA50	13	13	13	13
LA75	27	**25**	**25**	26
LGA75	29	28	**27**	28

Таблица Д.12. *E. coli str. O157H Sakai*-BEA1107

Assembly	Raw reads	Pollux	Coral	IonHammer
# misassemblies	2	1	**0**	**0**
# relocations	2	1	**0**	**0**
# translocations	0	0	0	0
# inversions	0	0	0	0
# misassembled contigs	2	1	**0**	**0**
Misassembled contigs length	252939	6225	**0**	**0**
# local misassemblies	21	17	20	**13**
# unaligned mis. contigs	0	0	0	0
# mismatches	**550**	754	585	691
# indels	**79**	100	878	89
# indels ($\leq$ 5 bp)	**75**	98	872	87
# indels ($>$ 5 bp)	4	**2**	6	**2**
Indels length	204	219	1075	**135**

Таблица Д.13. *E. coli str. O157H Sakai*-BEA1108

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# contigs (≥ 0 bp)	421	420	409	442	408
# contigs (≥ 1000 bp)	144	152	146	145	144
# contigs (≥ 5000 bp)	73	76	70	74	72
# contigs (≥ 10000 bp)	59	58	58	59	57
# contigs (≥ 25000 bp)	45	45	47	48	46
# contigs (≥ 50000 bp)	27	26	27	26	27
Total length (≥ 0 bp)	5355051	5352619	5357564	**5365357**	5358631
Total length (≥ 1000 bp)	5254237	5252490	**5262141**	5261677	5258790
Total length (≥ 5000 bp)	5090880	5088288	5099534	**5109701**	5105803
Total length (≥ 10000 bp)	4999466	4966589	**5015658**	5009310	5005003
Total length (≥ 25000 bp)	4753637	4722478	**4831798**	4822686	4818568
Total length (≥ 50000 bp)	4114849	4042249	4117211	4015561	**4119157**
# contigs	221	229	219	220	223
Largest contig	374895	374888	**375414**	374895	374892
Total length	5304973	5303739	5311218	**5311902**	5311861
Reference length	5498450	5498450	5498450	5498450	5498450
GC (%)	50.26	50.27	50.26	50.27	50.27
Reference GC (%)	50.54	50.54	50.54	50.54	50.54
N50	148479	148523	**149015**	148478	148479
NG50	146450	146448	**149015**	146445	146452
N75	71897	65712	66676	65712	**72405**
NG75	**49946**	44153	49642	44152	49577
L50	12	12	12	12	12
LG50	13	13	**12**	13	13
L75	**25**	26	**25**	26	**25**
LG75	**28**	**28**	**28**	29	**28**
# misassemblies	1	**0**	2	**0**	**0**
# misassembled contigs	1	**0**	2	**0**	**0**
Misassembled contigs length	4074	**0**	647303	**0**	**0**
# local misassemblies	18	**15**	**15**	16	16
# unaligned mis. contigs	0	0	0	0	0
# unaligned contigs	3 + 3 part	3 + 2 part	**1 + 3 part**	**1 + 3 part**	**1 + 3 part**
Unaligned length	93160	93167	93171	93090	**93058**
Genome fraction (%)	94.700	94.688	94.619	**94.845**	94.825
Duplication ratio	**1.001**	**1.001**	1.003	**1.001**	**1.001**
# N's per 100 kbp	0.00	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	12.64	12.39	15.55	**10.85**	12.14
# indels per 100 kbp	**3.15**	3.27	126.07	3.43	3.20
Largest alignment	374895	374888	**375414**	374895	374892
Total aligned length	5210492	5209413	5214834	**5217698**	5217675
NA50	148479	148523	**149015**	148478	148479
NGA50	145450	145448	**145646**	145446	145452
NA75	71897	65712	65787	65712	**72405**
NGA75	**49944**	44153	49642	44152	49577
LA50	12	12	12	12	12
LGA50	13	13	13	13	13
LA75	**25**	26	26	26	**25**
LGA75	**28**	**28**	**28**	29	**28**

Таблица Д.14. *E. coli str. O157H Sakai*-BEA1108

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# misassemblies	1	**0**	2	**0**	**0**
# relocations	1	**0**	2	**0**	**0**
# translocations	0	0	0	0	0
# inversions	0	0	0	0	0
# misassembled contigs	1	**0**	2	**0**	**0**
Misassembled contigs length	4074	**0**	647303	**0**	**0**
# local misassemblies	18	**15**	**15**	16	16
# unaligned mis. contigs	0	0	0	0	0
# mismatches	658	645	809	**566**	633
# indels	**164**	170	6559	179	167
# indels ($\leq$ 5 bp)	**160**	166	6552	174	163
# indels ($>$ 5 bp)	**4**	**4**	7	5	**4**
Indels length	297	308	7191	298	**296**

Таблица Д.15. *E. coli str. O157H Sakai*-C23

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# contigs (≥ 0 bp)	382	405	383	439	418
# contigs (≥ 1000 bp)	144	143	149	138	141
# contigs (≥ 5000 bp)	71	75	74	74	73
# contigs (≥ 10000 bp)	55	59	58	56	57
# contigs (≥ 25000 bp)	45	49	49	47	48
# contigs (≥ 50000 bp)	30	29	29	29	29
Total length (≥ 0 bp)	5350715	5350609	5348684	**5365125**	5355448
Total length (≥ 1000 bp)	5262004	5251202	5263170	**5266030**	5257021
Total length (≥ 5000 bp)	5100410	5100430	5106438	**5126515**	5109670
Total length (≥ 10000 bp)	4986006	4989504	4995782	**5000422**	4998511
Total length (≥ 25000 bp)	4806428	4828350	4846153	**4850697**	4844648
Total length (≥ 50000 bp)	**4240809**	4126510	4121147	4183686	4124393
# contigs	215	216	207	209	208
Largest contig	351990	351989	**352405**	351991	351990
Total length	5310318	5302289	5302368	**5314082**	5303571
Reference length	5498450	5498450	5498450	5498450	5498450
GC (%)	50.27	50.27	50.27	50.28	50.27
Reference GC (%)	50.54	50.54	50.54	50.54	50.54
N50	146445	146443	**146669**	146445	146442
NG50	141233	132638	132840	**142416**	132981
N75	**72090**	66171	65782	66757	66137
NG75	**65594**	53978	49990	58040	53977
L50	12	12	12	12	12
LG50	13	13	13	13	13
L75	**26**	27	27	**26**	27
LG75	**28**	29	30	**28**	29
# misassemblies	0	0	0	0	0
# misassembled contigs	0	0	0	0	0
Misassembled contigs length	0	0	0	0	0
# local misassemblies	21	**19**	20	20	**19**
# unaligned mis. contigs	**0**	**0**	1	**0**	**0**
# unaligned contigs	2 + 2 part	2 + 2 part	**1 + 3 part**	3 + 2 part	**1 + 3 part**
Unaligned length	93141	93140	93101	**92866**	93059
Genome fraction (%)	94.796	94.670	94.546	**94.883**	94.686
Duplication ratio	**1.001**	**1.001**	1.002	**1.001**	**1.001**
# N's per 100 kbp	0.00	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	13.64	12.62	15.41	11.98	**11.91**
# indels per 100 kbp	**4.03**	4.25	105.74	4.12	4.21
Largest alignment	351990	351989	**352405**	351991	351990
Total aligned length	5216013	5208050	5207864	**5220106**	5209399
NA50	141233	145443	**145647**	145445	145442
NGA50	141233	132592	132782	**142358**	132923
NA75	**72090**	66171	65782	66757	66137
NGA75	**65594**	53978	49988	58039	49941
LA50	13	**12**	**12**	**12**	**12**
LGA50	13	13	13	13	13
LA75	**26**	27	27	**26**	27
LGA75	**28**	29	30	**28**	30

Таблица Д.16. *E. coli str. O157H Sakai*-C23

Assembly	Raw reads	Pollux	Coral	Fiona	IonHammer
# misassemblies	0	0	0	0	0
# relocations	0	0	0	0	0
# translocations	0	0	0	0	0
# inversions	0	0	0	0	0
# misassembled contigs	0	0	0	0	0
Misassembled contigs length	0	0	0	0	0
# local misassemblies	21	**19**	20	20	**19**
# unaligned mis. contigs	**0**	**0**	1	**0**	**0**
# mismatches	711	657	801	625	**620**
# indels	**210**	221	5497	215	219
# indels ($\leq$ 5 bp)	**206**	217	5492	211	215
# indels ($>$ 5 bp)	**4**	**4**	5	**4**	**4**
Indels length	**343**	371	5774	348	359

Таблица Д.17. *E. coli str. O157H Sakai*-C24

Assembly	Raw reads	Pollux	Coral	IonHammer
# contigs (≥ 0 bp)	445	422	430	450
# contigs (≥ 1000 bp)	152	143	142	149
# contigs (≥ 5000 bp)	75	76	71	75
# contigs (≥ 10000 bp)	57	55	54	58
# contigs (≥ 25000 bp)	46	45	45	46
# contigs (≥ 50000 bp)	28	27	28	26
Total length (≥ 0 bp)	5363630	5359567	**5375630**	5367370
Total length (≥ 1000 bp)	5264287	5258541	**5280963**	5264304
Total length (≥ 5000 bp)	5099227	5109710	**5132594**	5109791
Total length (≥ 10000 bp)	4975985	4960650	**5013090**	4988239
Total length (≥ 25000 bp)	4793238	4786028	**4872834**	4787372
Total length (≥ 50000 bp)	4136957	4131980	**4259796**	4052240
# contigs	216	216	214	223
Largest contig	375364	375363	**376363**	375706
Total length	5309397	5310467	**5329121**	5315505
Reference length	5498450	5498450	5498450	5498450
GC (%)	50.28	50.28	50.28	50.28
Reference GC (%)	50.54	50.54	50.54	50.54
N50	146452	148893	**149268**	149016
NG50	146012	146450	**147071**	146793
N75	66756	72646	**73059**	66136
NG75	54771	66172	**72809**	46392
L50	12	12	12	12
LG50	13	13	13	13
L75	26	**25**	**25**	26
LG75	28	27	**26**	28
# misassemblies	**0**	1	**0**	**0**
# misassembled contigs	**0**	1	**0**	**0**
Misassembled contigs length	**0**	6120	**0**	**0**
# local misassemblies	22	20	22	**19**
# unaligned mis. contigs	0	0	0	0
# unaligned contigs	9 + 3 part	10 + 2 part	**6 + 4 part**	9 + 3 part
Unaligned length	97647	97579	**97342**	97473
Genome fraction (%)	94.695	94.724	**94.917**	94.820
Duplication ratio	**1.001**	**1.001**	1.002	**1.001**
# N's per 100 kbp	0.00	0.00	0.00	0.00
# mismatches per 100 kbp	**13.04**	13.46	13.55	13.43
# indels per 100 kbp	**2.15**	2.30	148.15	2.49
Largest alignment	375364	375363	**376363**	375706
Total aligned length	5210016	5211726	**5230664**	5216795
NA50	145969	148893	**149268**	149016
NGA50	145453	145451	**146064**	145835
NA75	66756	72646	**73059**	66136
NGA75	54769	66172	**72809**	46392
LA50	12	12	12	12
LGA50	13	13	13	13
LA75	26	**25**	**25**	26
LGA75	28	27	**26**	28

Таблица Д.18. *E. coli str. O157H Sakai*-C24

Assembly	Raw reads	Pollux	Coral	IonHammer
# misassemblies	**0**	1	**0**	**0**
# relocations	**0**	1	**0**	**0**
# translocations	0	0	0	0
# inversions	0	0	0	0
# misassembled contigs	**0**	1	**0**	**0**
Misassembled contigs length	**0**	6120	**0**	**0**
# local misassemblies	22	20	22	**19**
# unaligned mis. contigs	0	0	0	0
# mismatches	**679**	701	707	700
# indels	**112**	120	7732	130
# indels ($\leq$ 5 bp)	**109**	117	7727	127
# indels ($>$ 5 bp)	**3**	**3**	5	**3**
Indels length	**238**	260	8455	254